Technische Universität Ilmenau
Fakultät für Informatik und Automatisierung
Institut für Theoretische Informatik
Fachgebiet Automaten und Logik

Bachelorarbeit

Analyse der Entscheidbarkeit diverser Probleme in automatischen Graphen

Vorgelegt von:
Chris Köcher

Betreuer:	Prof. Dr. Dietrich Kuske
2. Gutachter:	M. Sc. Martin Huschenbett
Studiengang:	Informatik 2011
Matrikel-Nr.:	48203
Eingereicht:	Ilmenau, den 10. September 2014

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Motivation und Zielstellung

In den Anwendungsgebieten der Informatik und Mathematik stellt sich häufig heraus, dass verschiedenste Probleme sich auf eines der klassischen Graphprobleme reduzieren lassen. Hierzu soll das folgende beispielhafte Szenario aus [Mar04] betrachtet werden:

Gegeben sei eine Menge von Prozessen und eine Menge von zur Verfügung stehenden Zeitfenstern. Jeweils zwei Prozesse können in Konflikt stehen und deshalb nicht gleichzeitig ausgeführt werden. Können diese Prozesse in den zur Verfügung stehenden Zeitfenstern ausgeführt werden, ohne dass zwei in Konflikt stehende Prozesse gleichzeitig abgearbeitet werden?

Dies lässt sich nun wie folgt als Färbungsproblem von Graphen modellieren: Die Knotenmenge ist die Menge der Prozessoren. Eine Kante zwischen zwei Knoten wird eingefügt, wenn die zugeordneten Prozesse miteinander in Konflikt stehen. Die zur Verfügung stehenden Zeitfenster entsprechen dann den verfügbaren Farben. Können die Knoten des Graphen so gefärbt werden, sodass keine zwei benachbarten Knoten die gleiche Farbe besitzen?

Für endliche Graphen ist dieses Problem entscheidbar, aber für mehr als zwei Farben NP-vollständig. Es stellt sich hierbei die vollkommen natürliche Frage, wie sich das Graphfärbungsproblem und auch andere Graphprobleme in unendlichen Graphen verhalten. Hirst und Harel haben in [HH96] eine Reihe solcher NP-vollständiger Graphprobleme auf rekursive Graphen (d.h. Knoten- und Kantenmenge können von Turingmaschinen entschieden werden) ausgeweitet und hierfür die Σ_1^1-Vollständigkeit festgestellt.

Da in rekursiven Strukturen bekanntlich die Menge der erfüllbaren prädikatenlogischen Formeln erster Stufe im Allgemeinen unentscheidbar ist (z.B. für $(\mathbb{N}, +, \cdot)$, siehe [Göd31]), wurde infolgedessen die Klasse der automatischen Strukturen (dies sind Strukturen, deren Universum und Relationen von endlichen Automaten akzeptiert werden können) untersucht. Khoussainov und Nerode haben in [KN95] für diese Klasse die Entscheidbarkeit der erfüllbaren Formeln der Prädikatenlogik erster Stufe festgestellt. Es wurden des Weiteren mit $\mathsf{FO}[\exists^\infty]$, $\mathsf{FO}[\exists^{\mathrm{mod}}]$ und $\mathsf{FO}[\exists^{\mathrm{ram}}]$ verschiedene Erweiterungen der Prädikatenlogik erster Stufe untersucht und deren Entscheidbarkeiten in automatischen Strukturen nachgewiesen [BG00, Rub08]. In [Kus09] sowie [KL10] wurde sogar die Entscheidbarkeit der sogenannten FSO-Logik, welche eine Einschränkung der Prädikatenlogik zweiter Stufe darstellt (Variablen zweiter Stufe dürfen relativ zum bindenden Quantoren nicht positiv in einer Formel vorkommen), bewiesen.

Für einige der Graphprobleme aus [HH96] wurde bereits die Entscheidbarkeit mithilfe der FSO-Logik oder aber auch die Unentscheidbarkeit für automatische Graphen nachgewiesen. In dieser Arbeit sollen nun eine Menge weiterer dieser Probleme auf ihre Entscheidbarkeit hin untersucht werden und in der arithmetischen bzw. analytischen Hierarchie eingeordnet werden. Für einige dieser Probleme stellt sich hierbei die Entscheidbarkeit heraus. Einige weitere Probleme dagegen sind im automatischen wie im rekursiven Fall Σ_1^1-vollständig. Des Weiteren wird gezeigt, dass das oben genannte Graphfärbungsproblem im automatischen Fall zwar nicht entscheidbar, aber mit Π_1^0 vollständig für eine niedrige Stufe der arithmetischen Hierarchie ist, während dies im rekursiven Fall Σ_1^1-vollständig ist.

1.2 Aufbau der Arbeit

In Kapitel 2 werden einige wichtige Grundlagen wie die Definition der arithmetischen und analytischen Hierarchie und weitere aus den Bereichen der Berechenbarkeitstheorie aufgelistet. Im Weiteren werden mit der Definition von automatischen Strukturen und dem Fundamentalsatz über automatische Strukturen auch die wichigsten Aussagen aus der Automatentheorie angegeben, die in den folgenden Kapiteln benötigt werden.

In Kapitel 3 wird zunächst die Definition der FSO-Logik wiederholt und die wichtigsten Aussagen zum Model-Checking-Problem dieser Logik aus [KL10] angegeben. Am Ende des Kapitels wird mithilfe dieser Aussage die Entscheidbarkeit verschiedener Probleme, wie dem Mengenüberdeckungsproblem (hier $\overline{\text{SET COVER}}^{\text{aut}}$ genannt) oder dem Mengenpackungsproblem (hier SET PACKING$^{\text{aut}}$ genannt), gezeigt.

Kapitel 4 beschäftigt sich mit der Σ_1^1-Vollständigkeit des Exakten Überdeckungsproblems EXACT COVER$^{\text{aut}}$, welches aus der Σ_1^1-Vollständigkeit der Existenz unendlicher Pfade in automatischen Graphen folgt.

Eine natürliche Erweiterung des Erfüllbarkeitsproblems aussagenlogischer Formeln in konjunktiver Normalform, welches hier als Graphproblem modelliert wird, wird in Kapitel 5 mit dem Problem SAT$^{\text{aut}}$ untersucht. Hierfür wird zunächst die Σ_1^1-Vollständigkeit dieses Problems durch Reduktion von EXACT COVER$^{\text{aut}}$ aus Kapitel 4 festgestellt. Im Weiteren wird dieses Problem noch auf zwei verschiedene Arten bezüglich der unendlichen Klauseln eingeschränkt: Besitzt die Formel keine unendlichen Klauseln, so kann mithilfe des Endlichkeitssatzes der Aussagenlogik und des Post'schen Korrespondenzproblems PCP die Π_1^0-Vollständigkeit gezeigt werden. Besitzt die Formel dagegen nur endlich viele unendliche Klauseln, so lässt sich mit der vorigen Aussage und einer Variante des PCP die Σ_2^0-Vollständigkeit zeigen.

Schließlich wird in Kapitel 6 noch das Graphfärbungsproblem COLOR$^{\text{aut}}$ untersucht, für welches sich die Π_1^0-Vollständigkeit herausstellt. Es wird gezeigt, dass COLOR$^{\text{aut}}$ mit unendlicher Farbmenge sogar entscheidbar ist und eine Färbung, welche im Allgemeinen jedoch nicht regulär ist, berechnet werden kann. Zudem wird ebenfalls mithilfe des PCP gezeigt, dass COLOR$^{\text{aut}}$ mit endlicher Farbmenge und sogar mit nur zwei Farben Π_1^0-vollständig ist.

Kapitel 2

Grundlagen

Auf den folgenden Seiten dieses Kapitels sollen einige wichtige Definitionen und Sätze aus der Berechenbarkeits- und Automatentheorie sowie der Logik angegeben werden.

2.1 Berechenbarkeitstheorie

Eine **Turingmaschine** ist ein Tupel $\mathcal{M} = (Q, \Sigma, \Gamma, \Delta, q_0, \square, F)$, wobei Q eine endliche Menge von *Zuständen*, Σ das *Eingabealphabet*, Γ das *Bandalphabet*, $\Delta \subseteq Q \times \Gamma \times Q \times \Gamma \times \{L, R, N\}$ eine Menge von *Transitionen*, $q_0 \in Q$ der *Startzustand*, $\square \in \Gamma \setminus \Sigma$ das *Leerzeichen* und $F \subseteq Q$ eine Menge von *Finalzuständen* ist. Eine **Konfiguration** ist ein Wort $k \in \Gamma^* Q \Gamma^+$. Die **Transitionsrelation** $\vdash_{\mathcal{M}}$ ist wie folgt definiert:

- $vqaw \vdash_{\mathcal{M}} vpbw$, falls $(q, a, p, b, N) \in \Delta$ und $v, w \in \Gamma^*$,
- $vqaw \vdash_{\mathcal{M}} vbpw'$, falls $(q, a, p, b, R) \in \Delta$, $v, w, w' \in \Gamma^*$ und $w = w' \neq \varepsilon$ oder $w = \varepsilon$ und $w' = \square$,
- $vqaw \vdash_{\mathcal{M}} v'pcbw$, falls $(q, a, p, b, L) \in \Delta$, $c \in \Gamma$, $v, v', w \in \Gamma^*$ und $v = v'c$ oder $v = v' = \varepsilon$ und $c = \square$.

Eine Turingmaschine $\mathcal{M}$ **akzeptiert** ein Wort $w \in \Sigma^*$, wenn es eine Konfiguration $k = uqv$ mit $q \in F$ gibt, sodass $q_0 w \square \vdash_{\mathcal{M}}^* k$ und für alle Konfigurationen k' mit $k \vdash_{\mathcal{M}} k'$ gilt: $k = k'$. Die von $\mathcal{M}$ **akzeptierte Sprache $L(\mathcal{M})$** ist dann die folgende Menge:

$$L(\mathcal{M}) := \{w \in \Sigma^* \mid \mathcal{M} \text{ akzeptiert } w\}.$$

Eine **Mehrband-Turingmaschine** mit $n \geq 1$ Bändern ist ein Tupel $\mathcal{M} = (Q, \Sigma, \Gamma, \Delta, q_0, \square, F)$, wobei Q, Σ, Γ, q_0, $\square$ und F wie für Turingmaschinen definiert sind und $\Delta \subseteq Q \times \Gamma^n \times Q \times (\Gamma \times \{L, R, N\})^n$ gilt. Konfigurationen und akzeptierte Sprache von Mehrband-Turingmaschinen sind analog zu den Äquivalenten in Turingmaschinen definiert. Eine **Orakel-Turingmaschine** ist dann ein Tupel $\mathcal{M} = (Q, \Sigma, \Gamma, \Delta, q_0, \square, F, A)$, wobei $A \subseteq \Gamma^*$ das *Orakel* ist und $(Q, \Sigma, \Gamma, \Delta, q_0, \square, F)$ eine Mehrband-Turingmaschine mit zusätzlichen Transitionen der nachfolgenden Form ist: „Falls für das Wort $w \in \Gamma^*$, das auf Band i steht, $w \in A$ gilt, so gehe in Zustand $q \in Q$, sonst in Zustand $p \in Q$."

Für die von $\mathcal{M}$ akzeptierte Sprache schreibt man dann $L(\mathcal{M}^A)$.

Die **Arithmetische Hierarchie** ist nun wie folgt definiert:

$$\begin{aligned}
\boldsymbol{\Sigma_1^0} &:= \{L(\mathcal{M}) \mid \mathcal{M} \text{ ist Turingmaschine}\},\\
\boldsymbol{\Pi_1^0} &:= \{\overline{L(\mathcal{M})} \mid \mathcal{M} \text{ ist Turingmaschine}\},\\
\boldsymbol{\Delta_1^0} &:= \Sigma_1^0 \cap \Pi_1^0,\\
\boldsymbol{\Sigma_{n+1}^0} &:= \{L(\mathcal{M}^A) \mid \mathcal{M} \text{ ist Orakel-Turingmaschine mit Orakel } A \in \Sigma_n^0\},\\
\boldsymbol{\Pi_{n+1}^0} &:= \{\overline{L(\mathcal{M}^A)} \mid \mathcal{M} \text{ ist Orakel-Turingmaschine mit Orakel } A \in \Sigma_n^0\},\\
\boldsymbol{\Delta_{n+1}^0} &:= \Sigma_{n+1}^0 \cap \Pi_{n+1}^0.
\end{aligned}$$

Σ_1^0 ist also die Klasse der semi-entscheidbaren Sprachen, Π_1^0 die Klasse der co-semi-entscheidbaren Sprachen und Δ_1^0 die Klasse der entscheidbaren Sprachen. Zudem lässt sich die Klasse Σ_n^0 auch charakterisieren als die Klasse aller Sprachen $A \subseteq \Sigma^*$ mit:

$$A = \{y \in \Sigma^* \mid \exists x_1 \forall x_2 \exists x_3 \ldots Q_n x_n \colon R(y, x_1, \ldots, x_n)\},$$

wobei Σ ein Alphabet, $R \subseteq (\Sigma^*)^{n+1}$ ein entscheidbares Prädikat und $Q_n = \exists$, falls n ungerade, und $Q_n = \forall$ sonst ist.

Als erste Stufe der **Analytischen Hierarchie** wird zudem noch die Klasse $\boldsymbol{\Sigma_1^1}$ aller Probleme der Form

$$\{y \in \Sigma^* \mid \exists X \colon \phi(X, y)\}$$

betrachtet, wobei ϕ eine arithmetische FO-Formel ist, in der $X \subseteq \Sigma^*$ als unäres Prädikat verwendet werden kann.

Ein Problem A heißt **$\mathcal{K}$-vollständig** für eine Klasse $\mathcal{K}$, falls $A \in \mathcal{K}$ und A $\mathcal{K}$-hart ist, d.h. wenn alle Probleme $B \in \mathcal{K}$ auf A reduziert werden können.

Weitere Details zur arithmetischen und analytischen Hierarchie sowie der Berechenbarkeitstheorie im Allgemeinen können in [Koz06] und [Rog67] gefunden werden.

2.2 Automatentheorie

Ein **(endlicher) Automat** ist ein Tupel $\mathcal{A} = (Q, \Sigma, I, \Delta, F)$, wobei Q eine endliche Menge von *Zuständen*, Σ das *Eingabealphabet*, $I \subseteq Q$ die Menge der *Initialzustände*, $\Delta \subseteq Q \times \Delta \times Q$ eine Menge von *Transitionen* und $F \subseteq Q$ die Menge der *Finalzustände* ist. Eine **Konfiguration** ist ein Wort $k \in Q\Sigma^*$. Die **Transitionsrelation $\vdash_{\mathcal{A}}$** ist definiert durch $qaw \vdash_{\mathcal{A}} pw$, falls $(q, a, p) \in \Delta$ und $w \in \Sigma^*$ gilt. Die von $\mathcal{A}$ **akzeptierte Sprache $L(\mathcal{A})$** ist dann definiert durch:

$$L(\mathcal{A}) := \{w \in \Sigma^* \mid q_i w \vdash_{\mathcal{A}}^* q_f, q_i \in I, q_f \in F\}.$$

Eine Sprache heißt **regulär**, wenn sie von einem endlichen Automaten akzeptiert wird.

Mithilfe der folgenden beiden Definitionen sollen nun nicht nur einfache Wortsprachen von endlichen Automaten akzeptiert werden, sondern auch Relationen der Stelligkeit $n \geq 2$. Hierin wird die Abbildung

$$|.| \colon \Sigma^* \to \mathbb{N} \colon a_1 \ldots a_n \mapsto n$$

für $a_1, \ldots, a_n \in \Sigma$ verwendet, d.h. $|w|$ ist die Länge es Wortes $w \in \Sigma^*$.

Definition 2.1. Für ein Alphabet Σ sei $\boldsymbol{\Sigma_\diamond} := \Sigma \uplus \{\diamond\}$. Die **Konvolution** eines Tupels $(w_1, \ldots, w_n) \in (\Sigma^*)^n$ ist die Zeichenkette $\boldsymbol{\otimes(w_1, \ldots, w_n)} = a_1 \ldots a_l \in ((\Sigma_\diamond)^n)^*$ der Länge $l = \max\{|w_i| \mid 1 \leq i \leq n\}$, wobei sich für alle $1 \leq k \leq l$ das Symbol $a_k = (b_1, \ldots, b_n) \in (\Sigma_\diamond)^n$ wie folgt ergibt: Für alle $1 \leq i \leq n$ gilt:

$$b_i = \begin{cases} c_{i,k} & \text{, falls } k \leq |w_i| \text{ und } w_i = c_{i,1}c_{i,2}\ldots c_{i,|w_i|} \\ \diamond & \text{, falls } k > |w_i|. \end{cases}$$

Die **Konvolution** einer Relation $R \subseteq (\Sigma^*)^n$ ist die Menge $\boldsymbol{\otimes R} \subseteq ((\Sigma_\diamond)^n)^*$ der Konvolutionen aller Tupel in R, also:

$$\otimes R = \{\otimes(w_1, \ldots, w_n) \in ((\Sigma_\diamond)^n)^* \mid (w_1, \ldots, w_n) \in R\}.$$

Definition 2.2. Ein ***n*-Band-Automat** über Σ ist ein endlicher Automat über dem Alphabet $(\Sigma_\diamond)^n$. Die durch einen n-Band-Automaten $\mathcal{A}$ **definierte Relation** ist:

$$\boldsymbol{R(\mathcal{A})} := \{(w_1, \ldots, w_n) \mid \otimes(w_1, \ldots, w_n) \in L(\mathcal{A})\}.$$

Eine n-stellige Relation $R \subseteq (\Sigma^*)^n$ heißt **automatisch**, falls ein n-Band-Automat $\mathcal{A}$ existiert mit $R(\mathcal{A}) = R$.

Für die Klasse der regulären Sprachen sowie für die Klasse der automatischen Relationen ist die folgende Aussage bekannt:

Proposition 2.3 (Abschlusseigenschaften [Sch92, KN95]). *Die Klasse der regulären Sprachen ist unter Boole'schen Operationen, Projektionen, Konkatenation sowie der Kleene-Operation effektiv abgeschlossen.*

Die Klasse der automatischen Relationen ist unter Boole'schen Operationen sowie Projektionen effektiv abgeschlossen.

2.3 Strukturen und Logik

Eine **Signatur** ist ein Tupel $\tau = (\mathcal{R}, \mathcal{C}, \mathrm{ar})$, wobei $\mathcal{R}$ eine endliche Menge von *Relationssymbolen*, $\mathcal{C}$ eine endliche Menge von *Konstantensymbolen* und $\mathrm{ar}\colon \mathcal{R} \to \mathbb{N}$ die *Stelligkeitsfunktion* ist. Eine ***τ*-Struktur** ist ein Tupel $\mathcal{S} = (S, (R^\mathcal{S})_{R\in\mathcal{R}}, (c^\mathcal{S})_{c\in\mathcal{C}})$, wobei S das *Universum*, $R^\mathcal{S} \subseteq S^{\mathrm{ar}(R)}$ für alle $R \in \mathcal{R}$ Relationen und $c^\mathcal{S} \in S$ für alle $c \in \mathcal{C}$ Konstanten sind.

Eine τ-Struktur $\mathcal{S} = (S, (R^\mathcal{S})_{R\in\mathcal{R}}, (c^\mathcal{S})_{c\in\mathcal{C}})$ heißt **rekursiv**, wenn S sowie $R^\mathcal{S}$ für alle $R \in \mathcal{R}$ entscheidbar sind.

Definition 2.4. Sei $\tau = (\mathcal{R}, \mathcal{C}, \mathrm{ar})$ eine Signatur. Eine **automatische Darstellung** ist ein Tupel $\mathcal{D} = (\Sigma, \mathcal{A}, (\mathcal{A}_R)_{R\in\mathcal{R}}, (c^\mathcal{D})_{c\in\mathcal{C}})$ mit den folgenden Eigenschaften:

- Σ ist ein Alphabet,
- $\mathcal{A}$ ist ein endlicher Automat über dem Alphabet Σ,
- für alle $R \in \mathcal{R}$ ist $\mathcal{A}_R$ ein $\mathrm{ar}(R)$-Band-Automat mit $R(\mathcal{A}_R) \subseteq L(\mathcal{A})^{\mathrm{ar}(R)}$ und
- für alle $c \in \mathcal{C}$ gilt $c^{\mathcal{D}} \in L(\mathcal{A})$.

Die von $\mathcal{D}$ **dargestellte Struktur** ist dann definiert als:

$$\boldsymbol{\mathcal{S}(\mathcal{D})} := (L(\mathcal{A}), (R(\mathcal{A}_R))_{R\in\mathcal{R}}, (c^{\mathcal{D}})_{c\in\mathcal{C}}).$$

Eine Struktur $\mathcal{S}$ heißt **automatisch**, falls eine automatische Darstellung $\mathcal{D}$ existiert mit $\mathcal{S} \cong \mathcal{S}(\mathcal{D})$.

Bemerkung 2.5. In den nachfolgenden Kapiteln wird häufig eine automatische Struktur $\mathcal{S}$ (über einem Alphabet Σ) als Eingabe für Probleme angegeben. Da diese Strukturen im Allgemeinen nicht endlich sind, können diese natürlich keine Eingabe eines Algorithmus sein. Stattdessen wird implizit angenommen, dass die Eingabe eine automatischen Darstellung $\mathcal{D}$ (mit Alphabet Σ) mit $\mathcal{S}(\mathcal{D}) \cong \mathcal{S}$ ist.

Eine **$\mathsf{FO}[\tau]$-Formel** ist eine Formel der Prädikatenlogik erster Stufe, in der ausschließlich Relationssymbole aus $\mathcal{R}$ mit in ar definierter Stelligkeit und Konstantensymbole aus $\mathcal{C}$ vorkommen (vgl. Abschnitt 3.1, Regeln (i), (ii), (iv)-(vi)).

Für eine $\mathsf{FO}[\tau]$-Formel $\phi(x_1, \ldots, x_n)$ mit den freien Variablen $x_1, \ldots, x_n$ und eine τ-Struktur $\mathcal{S}$ definiere die folgende Relation:

$$\boldsymbol{\phi^{\mathcal{S}}} := \{(s_1, \ldots, s_n) \in S^n \mid \mathcal{S} \models \phi[s_1, \ldots, s_n]\}.$$

Schließlich ist die folgende auch „*Fundamentalsatz über automatische Strukturen*“ genannte Aussage bekannt:

Satz 2.6 (Fundamentalsatz [KN95]). *Aus einer automatischen Darstellung $\mathcal{D}$ und einer FO-Formel $\phi(x_1, \ldots, x_n)$ kann ein Automat $\mathcal{A}$ konstruiert werden mit $L(\mathcal{A}) = \otimes\phi^{\mathcal{S}(\mathcal{D})}$.*

Für einen FO-Satz ϕ und eine automatische Strukur $\mathcal{S}$ kann also entschieden werden, ob $\mathcal{S} \models \phi$ gilt.

Diese Aussage wurde in [BG00, Rub08] analog für Formeln über der Signatur τ erweitert, in denen zusätzlich zur FO-Logik auch die Quantoren $\exists^{\infty}$, $\exists^{(p,q)}$ sowie $\exists^{k\text{-ram}}$ verwendet werden.

Für automatische Strukturen kann noch die folgende Variante des Pumping-Lemmas angegeben werden.

Lemma 2.7 ([Blu99, Korollar 5.2]). *Sei $\tau = (\mathcal{R}, \mathcal{C}, \mathrm{ar})$ eine Signatur. Sei zudem $\mathcal{D} = (\Sigma, \mathcal{A}, (\mathcal{A}_R)_{R\in\mathcal{R}}, (c^{\mathcal{D}})_{c\in\mathcal{C}})$ eine automatische Darstellung und sei $R \in \mathcal{R}$ mit $\mathrm{ar}(R) = k + l$ eine Relation aus τ, sodass für alle $\bar{x} \in L(\mathcal{A})^k$ nur endlich viele $\bar{y} \in L(\mathcal{A})^l$ mit $(\bar{x}, \bar{y}) \in R(\mathcal{A}_R)$ existieren. Dann existiert eine Konstante m, sodass für alle $(\bar{x}, \bar{y}) \in R(\mathcal{A}_R)$ gilt:*

$$\max\{|y_i| \mid 1 \leq i \leq l\} \leq m + \max\{|x_i| \mid 1 \leq i \leq k\}.$$

2.4 Graphentheorie

Ein **(gerichteter) Graph** ist eine Struktur über der Signatur

$$\boldsymbol{\gamma} := (\{E\}, \emptyset, \{(E, 2)\}).$$

In dieser Arbeit wird häufig auch der Begriff „Graph“ für Strukturen über Signaturen mit Konstanten und mehreren Relationssymbolen, deren Stelligkeit maximal 2 beträgt, verwendet.

Ein **ungerichteter Graph** $\mathcal{G}$ ist ein symmetrischer Graph, d.h. alle zweistelligen Relationen in $\mathcal{G}$ sind symmetrisch. Ein Graph $\mathcal{G} = (V, E)$ heißt zudem **bipartit**, falls es eine Partition $V = U_1 \uplus U_2$ gibt, sodass $E \subseteq U_1 \times U_2 \cup U_2 \times U_1$ gilt. Für einen solchen Graphen wird auch kurz (U_1, U_2, E) geschrieben.

Ein **Pfad** (oder **Weg**) in einem Graphen $\mathcal{G} = (V, E)$ ist eine Folge $(v_1, \ldots, v_n)$ mit $n \geq 2$ und $(v_i, v_{i+1}) \in E$ für alle $1 \leq i < n$. Ein Pfad $(v_1, \ldots, v_n)$ besitzt die **Länge** $n - 1$. Ein **Kreis** ist ein Pfad $(v_1, \ldots, v_n)$ mit $v_1 = v_n$.

Ein **Nachfolgerbaum** $\mathcal{T} = (V, E, r)$ ist ein kreisfreier, asymmetrischer Graph (V, E) (d.h. E ist eine asymmetrische Relation), sodass es keinen Knoten $v \in V$ gibt mit $(v, r) \in E$ und für jeden Knoten $v \in V \setminus \{r\}$ gibt es einen eindeutigen Pfad $(r, \ldots, v)$. Ein **Ordnungsbaum** ist die reflexiv-transitive Hülle eines Nachfolgerbaums. Damit ist in einem Ordnungsbaum $\mathcal{T} = (V, E, r)$ die Teilstruktur (V, E) eine partielle Ordnung.

Ein Graph $\mathcal{G}$ heißt **rekursiv** (resp. **automatisch**), wenn $\mathcal{G}$ eine rekursive (resp. automatische) Struktur ist.

Kapitel 3

FSO-Logik

Zu Beginn soll die Entscheidbarkeit einiger wichtiger Graphprobleme in automatischen Graphen gezeigt werden, die in rekursiven Graphen als Σ_1^1-vollständig bekannt sind. Hierzu soll zunächst die Prädikatenlogik erster Stufe um eingeschränkte Mengenquantoren erweitert werden. Anschließend wird gezeigt, dass das Model-Checking-Problem für diese FSO[1] genannte Logik und automatische Strukturen ebenfalls entscheidbar ist.

3.1 Definition FSO-Logik

Sei $\tau = (\mathcal{R}, \mathcal{C}, \mathrm{ar})$ eine Signatur. Zudem sei $\mathcal{V}_0 = \{x_i \mid i \in \mathbb{N}\}$ eine abzählbare Menge von *Individualvariablen* und $\mathcal{V}_1^k = \{X_i^k \mid i \in \mathbb{N}\}$ für $k \geq 1$ eine abzählbare Menge der k-stelligen *Relationsvariablen*. Die Menge $\mathbf{FSO}[\boldsymbol{\tau}]$ der FSO-Formeln über der Signatur τ ist die kleinste Menge, die die folgenden Bedingungen erfüllt:

(i) Sind $x, y \in \mathcal{V}_0$, so gilt $(x = y) \in \mathsf{FSO}[\tau]$.

(ii) Ist $R \in \mathcal{R}$ mit $\mathrm{ar}(R) = k$ und $x_1, \ldots, x_k \in \mathcal{V}_0$, so gilt $R(x_1, \ldots, x_k) \in \mathsf{FSO}[\tau]$.

(iii) Ist $X \in \mathcal{V}_1^k$ für ein $k \geq 1$ und $x_1, \ldots, x_k \in \mathcal{V}_0$, so gilt $X(x_1, \ldots, x_k) \in \mathsf{FSO}[\tau]$.

(iv) Sind $\psi, \chi \in \mathsf{FSO}[\tau]$, so auch $\psi \vee \chi \in \mathsf{FSO}[\tau]$.

(v) Ist $\psi \in \mathsf{FSO}[\tau]$, so auch $\neg\psi \in \mathsf{FSO}[\tau]$.

(vi) Ist $\psi \in \mathsf{FSO}[\tau]$ und $x \in \mathcal{V}_0$, so auch $\exists x\colon \psi \in \mathsf{FSO}[\tau]$.

(vii) Ist $\psi \in \mathsf{FSO}[\tau]$ und $x \in \mathcal{V}_0$, so auch $\exists^\infty x\colon \psi \in \mathsf{FSO}[\tau]$.

(viii) Ist $\psi \in \mathsf{FSO}[\tau]$, $x \in \mathcal{V}_0$, $0 \leq p < q$, so auch $\exists^{(p,q)} x\colon \psi \in \mathsf{FSO}[\tau]$.

(ix) Ist $\psi \in \mathsf{FSO}[\tau]$ und $X \in \mathcal{V}_1^k \setminus \mathrm{pos}(\psi)$, so auch $\exists X$ infinite$\colon \psi \in \mathsf{FSO}[\tau]$.

Die Mengen $\mathrm{pos}(\phi)$ und $\mathrm{neg}(\phi)$ sind hierbei die Mengen der positiv bzw. negativ vorkommenden freien Variablen zweiter Stufe in der $\mathsf{FSO}[\tau]$-Formel ϕ. Genauer lassen sich die Funktionen $\mathrm{pos}, \mathrm{neg}\colon \mathsf{FSO}[\tau] \to 2^{\bigcup_{k\geq 1} \mathcal{V}_1^k}$ wie folgt induktiv über den Aufbau einer $\mathsf{FSO}[\tau]$-Formel definieren:

(i) $\mathrm{pos}(x = y) = \mathrm{neg}(x = y) = \emptyset$ für $x, y \in \mathcal{V}_0$.

(ii) $\mathrm{pos}(R(x_1, \ldots, x_k)) = \mathrm{neg}(R(x_1, \ldots, x_k)) = \emptyset$ für $R \in \mathcal{R}$, $\mathrm{ar}(R) = k$, $x_1, \ldots, x_k \in \mathcal{V}_0$.

[1] FSO ist eine Abkürzung für „**F**ragment of **S**econd **O**rder Logic“.

(iii) $\mathrm{pos}(X(x_1,\ldots,x_k)) = \{X\}$, $\mathrm{neg}(X(x_1,\ldots,x_k)) = \emptyset$ für $k \geq 1$, $X \in \mathcal{V}_1^k$, $x_1,\ldots,x_k \in \mathcal{V}_0$.

(iv) $\mathrm{pos}(\psi \vee \chi) = \mathrm{pos}(\psi) \cup \mathrm{pos}(\chi)$, $\mathrm{neg}(\psi \vee \chi) = \mathrm{neg}(\psi) \cup \mathrm{neg}(\chi)$ für $\psi, \chi \in \mathsf{FSO}[\tau]$.

(v) $\mathrm{pos}(\neg\psi) = \mathrm{neg}(\psi)$, $\mathrm{neg}(\neg\psi) = \mathrm{pos}(\psi)$ für $\psi \in \mathsf{FSO}[\tau]$.

(vi) $\mathrm{pos}(\exists x\colon \psi) = \mathrm{pos}(\psi)$, $\mathrm{neg}(\exists x\colon \psi) = \mathrm{neg}(\psi)$ für $\psi \in \mathsf{FSO}[\tau]$, $x \in \mathcal{V}_0$.

(vii) $\mathrm{pos}(\exists^\infty x\colon \psi) = \mathrm{pos}(\psi)$, $\mathrm{neg}(\exists^\infty x\colon \psi) = \mathrm{neg}(\psi)$ für $\psi \in \mathsf{FSO}[\tau]$, $x \in \mathcal{V}_0$.

(viii) $\mathrm{pos}(\exists^{(p,q)} x\colon \psi) = \mathrm{pos}(\psi)$, $\mathrm{neg}(\exists^{(p,q)} x\colon \psi) = \mathrm{neg}(\psi)$ für $\psi \in \mathsf{FSO}[\tau]$, $x \in \mathcal{V}_0$, $0 \leq p < q$.

(ix) $\mathrm{pos}(\exists X \text{ infinite}\colon \psi) = \mathrm{pos}(\psi) \setminus \{X\}$, $\mathrm{neg}(\exists X \text{ infinite}\colon \psi) = \mathrm{neg}(\psi) \setminus \{X\}$ für $\psi \in \mathsf{FSO}[\tau]$, $X \in \mathcal{V}_1^k$, $k \geq 1$.

Die *Menge der freien Variablen* $\mathbf{fV}(\boldsymbol{\phi})$ einer FSO-Formel ϕ ist analog definiert zur Menge der freien Variablen in FO- oder SO-Formeln.

Es muss nun noch die Semantik der FSO-Formeln definiert werden. Sei dazu $\mathcal{S} = (S, (R^{\mathcal{S}})_{R\in\mathcal{R}}, (c^{\mathcal{S}})_{c\in\mathcal{C}})$ eine τ-Struktur. Eine *Belegung in* $\mathcal{S}$ ist eine Abbildung

$$\alpha\colon \mathcal{V}_0 \cup \bigcup_{k\geq 1} \mathcal{V}_1^k \to S \cup \bigcup_{k\geq 1} 2^{S^k}$$

mit $\alpha(x) \in S$ für alle $x \in \mathcal{V}_0$ und $\alpha(X) \subseteq S^k$ für alle $X \in \mathcal{V}_1^k$ und alle $k \geq 1$.

Sei nun $\mathcal{S}$ eine τ-Struktur, α eine Belegung in $\mathcal{S}$ und $\phi \in \mathsf{FSO}[\tau]$. Dann ist $\boldsymbol{\mathcal{S} \models_\alpha \phi}$ (sprich: „ϕ gilt in $\mathcal{S}$ unter der Belegung α“) wie folgt induktiv über den Aufbau von ϕ definiert:

(i) $\mathcal{S} \models_\alpha (x = y)$ gdw. $\alpha(x) = \alpha(y)$ für $x, y \in \mathcal{V}_0$.

(ii) $\mathcal{S} \models_\alpha R(x_1,\ldots,x_k)$ gdw. $(\alpha(x_1),\ldots,\alpha(x_k)) \in R^{\mathcal{S}}$ für $R \in \mathcal{R}$, $\mathrm{ar}(R) = k$, $x_1,\ldots,x_k \in \mathcal{V}_0$.

(iii) $\mathcal{S} \models_\alpha X(x_1,\ldots,x_k)$ gdw. $(\alpha(x_1),\ldots,\alpha(x_k)) \in \alpha(X)$ für $k \geq 1$, $X \in \mathcal{V}_1^k$, $x_1,\ldots,x_k \in \mathcal{V}_0$.

(iv) $\mathcal{S} \models_\alpha \psi \vee \chi$ gdw. $\mathcal{S} \models_\alpha \psi$ oder $\mathcal{S} \models_\alpha \chi$ für $\psi, \chi \in \mathsf{FSO}[\tau]$.

(v) $\mathcal{S} \models_\alpha \neg\psi$ gdw. nicht $\mathcal{S} \models_\alpha \psi$ für $\psi \in \mathsf{FSO}[\tau]$.

(vi) $\mathcal{S} \models_\alpha \exists x\colon \psi$ gdw. ein $a \in S$ existiert mit $\mathcal{S} \models_{\alpha[\frac{x}{a}]} \psi$[2] für $\psi \in \mathsf{FSO}[\tau]$, $x \in \mathcal{V}_0$.

(vii) $\mathcal{S} \models_\alpha \exists^\infty x\colon \psi$ gdw. unendlich viele $a \in S$ existieren mit $\mathcal{S} \models_{\alpha[\frac{x}{a}]} \psi$ für $\psi \in \mathsf{FSO}[\tau]$, $x \in \mathcal{V}_0$.

(viii) $\mathcal{S} \models_\alpha \exists^{(p,q)} x\colon \psi$ gdw. $|\{a \in S \mid \mathcal{S} \models_{\alpha[\frac{x}{a}]} \psi\}| \equiv p \pmod q$ für $\psi \in \mathsf{FSO}[\tau]$, $x \in \mathcal{V}_0$, $0 \leq p < q$.

(ix) $\mathcal{S} \models_\alpha \exists X \text{ infinite}\colon \psi$ gdw. ein $A \subseteq S^k$ existiert mit $|A| = \infty$ und $\mathcal{S} \models_{\alpha[\frac{X}{A}]} \psi$ für $\psi \in \mathsf{FSO}[\tau]$, $X \in \mathcal{V}_1^k$, $k \geq 1$.

[2] $\alpha[\frac{x}{a}]$ ist die Belegung, die x auf a abbildet und sonst überall mit α übereinstimmt.

Analog zur FO-Logik gilt auch in der FSO-Logik die folgende Aussage:

Lemma 3.1. *Sei $\mathcal{S}$ eine τ-Struktur, $\phi \in \mathsf{FSO}[\tau]$ und α, β Belegungen in $\mathcal{S}$, sodass $\alpha(x) = \beta(x)$ für alle $x \in \mathrm{fV}(\phi)$ gilt. Dann gilt:*

$$\mathcal{S} \models_\alpha \phi \Leftrightarrow \mathcal{S} \models_\beta \phi.$$

Damit kann kurz $\mathcal{S} \models \phi[\alpha(x_1), \ldots, \alpha(x_n)]$ für $\mathcal{S} \models_\alpha \phi$ geschrieben werden. Insbesondere schreibt man $\mathcal{S} \models \phi$ für einen Satz $\phi \in \mathsf{FSO}$.

Für $\phi \in \mathsf{FSO}[\tau]$ mit $\mathrm{fV}(\phi) = \{x_1, \ldots, x_n\} \subseteq \mathcal{V}_0$ und eine τ-Struktur $\mathcal{S}$ definiere $\boldsymbol{\phi^{\mathcal{S}}}$ analog zur Definition in Abschnitt 2.3 die folgende Relation:

$$\phi^{\mathcal{S}} := \{(s_1, \ldots, s_n) \in S^n \mid \mathcal{S} \models \phi[s_1, \ldots, s_n]\}.$$

3.2 Entscheidbarkeit in automatischen Strukturen

Im folgenden Abschnitt werden zwei wichtige Sätze von Kuske und Lohrey angegeben, die Aussagen zur Entscheidbarkeit der FSO-Logik treffen. Die Beweise hiervon werden in dieser Arbeit jedoch nicht angegeben und können stattdessen in [Kus09, KL10] nachgelesen werden.

Die erste Aussage betrifft die Entscheidbarkeit von FSO-Sätzen in automatischen Strukturen:

Satz 3.2 ([KL10, Theorem 5.2, Korollar 5.3]). *Aus einer automatischen Darstellung $\mathcal{D}$ und einer FSO-Formel $\phi(x_1, \ldots, x_n)$ mit $x_1, \ldots, x_n \in \mathcal{V}_0$ kann ein Automat $\mathcal{A}$ konstruiert werden mit $L(\mathcal{A}) = \otimes\phi^{\mathcal{S}(\mathcal{D})}$.*

Für einen FSO-Satz ϕ und eine automatische Strukur $\mathcal{S}$ kann also entschieden werden, ob $\mathcal{S} \models \phi$ gilt.

Die zweite Aussage erweitert Satz 3.2 wie folgt um FSO-Formeln mit einer freien Variablen zweiter Stufe.

Satz 3.3 ([KL10, Theorem 5.4]). *Aus einer automatischen Darstellung $\mathcal{D}$ und einer FSO-Formel $\phi(X)$ mit $X \in \mathcal{V}_1^k$, $k \geq 1$, $X \notin \mathrm{pos}(\phi)$ und*

$$\mathcal{S}(\mathcal{D}) \models \exists X \text{ infinite}\colon \phi$$

kann eine automatische Relation $R \subseteq L(\mathcal{A})^k$ berechnet werden, sodass R unendlich ist und $\mathcal{S} \models \phi[R]$ gilt.

3.3 Anwendungen

Mithilfe der Entscheidbarkeit der FSO-Logik in automatischen Strukturen sollen nun einige Entscheidungsprobleme für Graphen untersucht werden, die in rekursiven Graphen allesamt Σ_1^1-vollständig sind.

3.3.1 Unabhängige Knotenmengen

Als erste Anwendung der FSO-Logik soll das Problem IND SET$^\infty$ betrachtet werden, das wie folgt definiert ist:

Eingabe: Ein rekursiver Graph $\mathcal{G} = (V, E)$.
Frage: Existiert eine unendliche Menge $X \subseteq V$, sodass für alle Knoten $u, v \in X$ gilt: $(u, v) \notin E$.

Dieses Problem ist nach [HH96, Proposition 4] Σ_1^1-vollständig. Die automatische Version, welche im Weiteren IND SET$^{\text{aut}}$ genannt wird und nachfolgend definiert ist, ist dagegen entscheidbar:

Eingabe: Ein *automatischer* Graph $\mathcal{G} = (V, E)$.
Frage: Existiert eine unendliche Menge $X \subseteq V$, sodass für alle Knoten $u, v \in X$ gilt: $(u, v) \notin E$.

Korollar 3.4. *Das Problem* IND SET$^{\text{aut}}$ *ist entscheidbar. Existiert eine unendliche Menge unabhängiger Knoten, so existiert auch eine effektiv reguläre Menge unabhängiger Knoten.*

Beweis. Betrachte die folgende FSO-Formel:

$$\exists X \text{ infinite}\colon \forall x \forall y\colon ((X(x) \wedge X(y)) \rightarrow \neg E(x, y)).$$

Diese Formel überprüft genau die in IND SET$^{\text{aut}}$ genannte Frage. Nach Satz 3.2 ist dies für automatische Graphen entscheidbar.

Nach Satz 3.3 kann zudem für den automatischen Graphen $\mathcal{G} = (V, E)$ eine solche reguläre Menge $X \subseteq V$ unabhängiger Knoten berechnet werden. ∎

3.3.2 Unendliche Pfade in Ordnungsbäumen

Ein fundamentales Σ_1^1-vollständiges Problem ist die Frage nach der Existenz eines unendlichen Pfades in einem rekursiven Baum (siehe hierzu auch [Rog67, Theorem 16.XX]). Dies gilt sowohl für Nachfolgerbäume als auch für Ordnungsbäume. Es soll nun die Restriktion dieses Problems auf automatische Ordnungsbäume analysiert werden. Für diese ergibt sich die folgende Aussage:

Korollar 3.5 ([KRS05, Proposition 8.1]). *Es ist entscheidbar, ob ein automatischer Ordnungsbaum einen unendlichen Pfad besitzt.*

Beweis. Betrachte die folgende FSO-Formel:

$$\phi := \exists X \text{ infinite}\colon \forall x \forall y\colon ((X(x) \wedge X(y)) \rightarrow (E(x,y) \vee E(y,x))).$$

Diese Formel überprüft, ob es im automatischen Ordnungsbaum $\mathcal{T} = (V, E)$ eine unendliche Knotenmenge $X \subseteq V$ gibt, die linear geordnet ist. In einem Ordnungsbaum sind ausschließlich Pfade lineare Ordnungen. Demzufolge besitzt $\mathcal{T}$ genau dann eine unendliche lineare Teilordnung, wenn es einen unendlichen Pfad in $\mathcal{T}$ gibt. Nach Satz 3.2 kann für $\mathcal{T}$ entschieden werden, ob $\mathcal{T} \models \phi$ gilt. ∎

3.3.3 Mengenüberdeckungen

Ein weiteres Σ^1_1-vollständiges Problem ist $\overline{\text{Set Cover}}^{\infty}$ (siehe [HH96, Proposition 6(b)]), welches wie folgt definiert ist:

Eingabe: Ein rekursiver, bipartiter Graph $\mathcal{G} = (U, V, E)$ mit $E \subseteq U \times V$.
Frage: Gibt es eine Menge $X \subseteq V$ mit $V \setminus X$ unendlich, sodass für jedes $u \in U$ ein $v \in X$ existiert mit $(u, v) \in E$?

Die Intuition hinter dem Graphen $\mathcal{G} = (U, V, E)$ ist die folgende:

- U ist eine Menge von Elementen, die sogenannte *Grundmenge*.
- V ist eine Menge von Repräsentanten für Teilmengen von U.
- Es gelte $(u, v) \in E$ genau dann, wenn u in der Teilmenge, die v repräsentiert, enthalten ist.
- Die gesuchte Menge $X \subseteq V$ ist eine Menge von Teilmengen von U, deren Vereinigung die Menge U ist.

Damit modelliert $\overline{\text{Set Cover}}^{\infty}$ also die unendliche Version des aus [Kar72] bekannten Problems Set Cover in einem Graphen.

Die automatische Version $\overline{\text{Set Cover}}^{\text{aut}}$ ist dann die folgende:

Eingabe: Ein *automatischer*, bipartiter Graph $\mathcal{G} = (U, V, E)$ mit $E \subseteq U \times V$.
Frage: Gibt es eine Menge $X \subseteq V$ mit $V \setminus X$ unendlich, sodass für jedes $u \in U$ ein $v \in X$ existiert mit $(u, v) \in E$?

Für dieses Problem lässt sich die folgende Aussage relativ einfach zeigen:

Korollar 3.6 ([KL10, Korollar 5.6]). *Das Problem* $\overline{\text{SET COVER}}^{\text{aut}}$ *ist entscheidbar. Existiert eine solche Mengenüberdeckung, so existiert auch eine effektiv reguläre Mengenüberdeckung.*

Beweis. Betrachte die folgende FSO-Formel:

$$\exists X \text{ infinite}\colon (\forall v\colon X(v) \to V(v) \wedge \forall u \exists v\colon U(u) \to (\neg X(v) \wedge E(u,v))).$$

Diese Formel überprüft, ob ein unendliches Komplement $X \subseteq V$ einer solchen Mengenüberdeckung existiert. Nach Satz 3.2 ist dies für automatische Strukturen entscheidbar.

Nach Satz 3.3 kann eine reguläre Mengenüberdeckung $X \subseteq V$ berechnet werden. Nach den Abschlusseigenschaften regulärer Sprachen kann die gesuchte (reguläre) Menge $V \setminus X$ berechnet werden. ∎

3.3.4 Set Packing

Das vorerst letzte zu untersuchende Problem, dessen automatische Variante mittels FSO-Logik entscheidbar ist, ist SET PACKING$^{\infty}$, welches wie folgt definiert ist:

Eingabe: Ein rekursiver, bipartiter Graph $\mathcal{G} = (U, V, E)$, wobei für jedes $v \in V$ unendlich viele $u \in U$ mit $(u, v) \in E$ existieren.
Frage: Existiert eine unendliche Menge $X \subseteq V$, sodass für alle $v, v' \in X$ und $u \in U$ mit $(u, v), (u, v') \in E$ gilt $v = v'$?

Die Intuition hinter den Mengen U, V und E des Graphen $\mathcal{G} = (U, V, E)$ ist die gleiche wie für $\overline{\text{SET COVER}}^{\infty}$. Die gesuchte Menge $X \subseteq V$ ist dagegen eine Menge von Teilmengen von U, die allesamt paarweise disjunkt sind. Damit ist dies also tatsächlich die unendliche Version des Problems SET PACKING aus [Kar72].

In [Kar72, HH96] wurde gezeigt, dass SET PACKING$^{\infty}$ Σ^1_1-vollständig ist. Es soll deshalb nun die automatische Version SET PACKING$^{\text{aut}}$ untersucht werden:

Eingabe: Ein *automatischer*, bipartiter Graph $\mathcal{G} = (U, V, E)$, wobei für jedes $v \in V$ unendlich viele $u \in U$ mit $(u, v) \in E$ existieren.
Frage: Existiert eine unendliche Menge $X \subseteq V$, sodass für alle $v, v' \in X$ und $u \in U$ mit $(u, v), (u, v') \in E$ gilt $v = v'$?

Korollar 3.7. *Das Problem* SET PACKING$^{\text{aut}}$ *ist entscheidbar. Existiert eine solche Menge X, so existiert auch eine effektiv reguläre Menge.*

Beweis. Sei $\mathcal{G} = (U, V, E)$ ein automatischer, bipartiter Graph wie oben. Betrachte die folgende FSO-Formel:

$$\phi := \exists X \text{ infinite}\colon \forall u \forall v \forall v' \colon ((X(v) \wedge X(v') \wedge E(u, v) \wedge E(u, v')) \rightarrow v = v').$$

Es gilt: $\mathcal{G} \in$ Set Packing$^{\text{aut}}$ genau dann, wenn $\mathcal{G} \models \phi$. Letzteres ist nach Satz 3.2 entscheidbar.

Nach Satz 3.3 kann zudem aus ϕ eine Menge $X \subseteq V$ berechnet werden. ∎

Kapitel 4

Exakte Überdeckungen

In diesem Abschnitt soll ein weiteres Problem mit Mengenüberdeckungen betrachtet werden. Im Unterschied zu $\overline{\text{Set Cover}}^{\infty}$ aus Abschnitt 3.3.3 werden hier nur zweielementige Mengen betrachtet und die gewählten Mengen müssen paarweise disjunkt sein. Es wird mit diesem Problem also nach einer exakten Überdeckung bzw. einem perfekten Matching von V gesucht. Genauer ist das Exact Cover$^{\infty}$ genannte Problem wie folgt definiert:

Eingabe: Ein rekursiver, ungerichteter Graph $\mathcal{G} = (V, E)$.
Frage: Gibt es eine Menge $C \subseteq E$, sodass für jedes $v \in V$ *genau ein* $u \in V \setminus \{v\}$ existiert mit $(v, u) \in C$ oder $(u, v) \in C$?

Nach [HH96, Proposition 10] ist dieses Problem ebenfalls Σ_1^1-vollständig. Es soll nun wieder die folgende automatische Variante dieses Problems untersucht werden:

Eingabe: Ein *automatischer*, ungerichteter Graph $\mathcal{G} = (V, E)$.
Frage: Gibt es eine Menge $C \subseteq E$, sodass für jedes $v \in V$ *genau ein* $u \in V \setminus \{v\}$ existiert mit $(v, u) \in C$ oder $(u, v) \in C$?

Im Gegensatz zum entscheidbaren $\overline{\text{Set Cover}}^{\text{aut}}$ ist dieses Exact Cover$^{\text{aut}}$ genannte Problem ebenfalls Σ_1^1-vollständig. Um dies zu zeigen, wird die folgende Aussage verwendet, welche hier jedoch nicht bewiesen werden soll.

Satz 4.1 ([KL10, Theorem 3.6]). *Die Frage nach der Existenz eines unendlichen Pfades in automatischen Nachfolgerbäumen ist Σ_1^1-vollständig.*

Dieses Problem wird nun im folgenden Beweis auf Exact Cover$^{\text{aut}}$ reduziert. Dieser Beweis ist hierbei eine Variation des Beweises aus [HH96, Proposition 10].

Satz 4.2. *Das Problem* Exact Cover$^{\text{aut}}$ *ist Σ_1^1-vollständig.*

Beweis. Da jeder automatische Graph $\mathcal{G} = (V, E)$ auch rekursiv ist und EXACT COVER$^{\infty}$ in Σ_1^1 liegt, ist auch EXACT COVER$^{\text{aut}}$ in Σ_1^1. Es muss also nur noch die Σ_1^1-Härte gezeigt werden. Dies soll durch Reduktion des Existenzproblems unendlicher Pfade in automatischen Nachfolgerbäumen, welches nach Satz 4.1 Σ_1^1-vollständig ist, auf dieses Problem gezeigt werden.

Idee. Ersetze im Nachfolgerbaum $\mathcal{T} = (V, E, r)$ jeden Knoten außer der Wurzel $v \in V \setminus \{r\}$ durch zwei Kopien v und $v\#$ und verbinde diese mit einer Kante. Aus einer Kante $(u, v) \in E$ wird anschließend die neue Kante $(u, v\#)$. Eine exakte Überdeckung des konstruierten Graphen enthält dann die Kanten $(v_i, v_{i+1}\#)$ genau eines unendlichen Pfades $(v_0, v_1, \dots)$ im Nachfolgerbaum. Die restlichen Kanten in der Überdeckung haben die Form $(v, v\#)$.

Sei nun also $\mathcal{T} = (N, P, r)$ ein automatischer Nachfolgerbaum über dem Alphabet Σ. Konstruiere hieraus einen Graphen $\mathcal{X}(\mathcal{T}) = (V, E)$ mit:

- $V := N \cup (N \setminus \{r\})\{\#\}$ mit $\# \notin \Sigma$,
- $E := \{(u, v\#), (v\#, u) \mid (u, v) \in P\} \cup \{(u, u\#), (u\#, u) \mid u \in N \setminus \{r\}\}$.

In Abbildung 4.1 wird die Konstruktion $\mathcal{T} \mapsto \mathcal{X}(\mathcal{T})$ veranschaulicht.

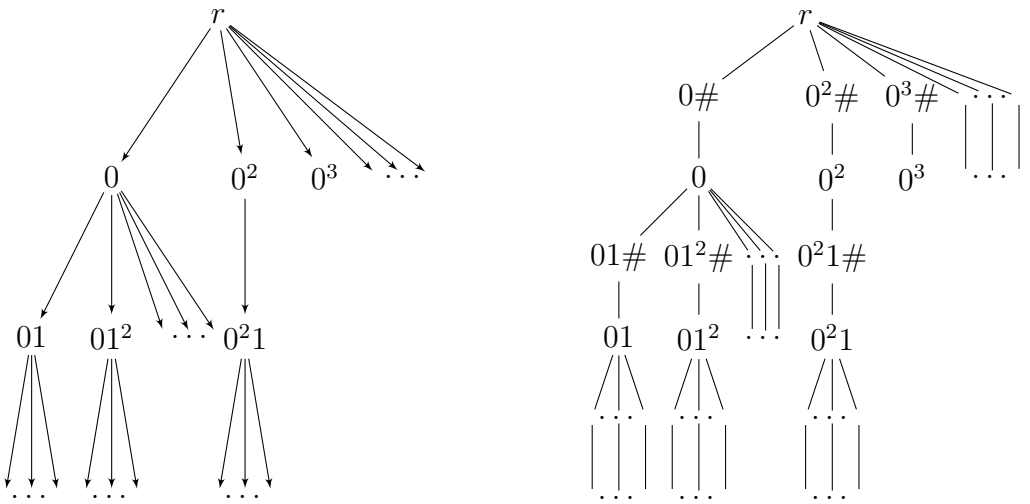

Abbildung 4.1. Links der Nachfolgerbaum $\mathcal{T} = (N, P, r)$, rechts der konstruierte Graph $\mathcal{X}(\mathcal{T}) = (V, E)$

Behauptung 1. Besitzt $\mathcal{T}$ einen unendlichen Pfad, so gibt es in $\mathcal{X}(\mathcal{T})$ eine exakte Überdeckung.
Beweis. Besitze $\mathcal{T}$ nun einen unendlichen Pfad. Sei $(v_1, v_2, v_3, \dots)$ ein solcher Pfad mit o.B.d.A. $v_1 = r$. Dann ist die folgende Menge eine exakte Überdeckung:

$$C := \{(v_i, v_{i+1}\#) \mid i \geq 1\} \cup \{(u, u\#) \mid u \notin \{v_1, v_2, \dots\}\}.$$ q.e.d.

Behauptung 2. Besitzt $\mathcal{X}(\mathcal{T})$ eine exakte Überdeckung, so besitzt $\mathcal{T}$ einen unendlichen Pfad.

Beweis. Sei nun $C \subseteq E$ eine exakte Überdeckung von $\mathcal{X}(\mathcal{T})$. Dann konstruiere nun induktiv über $i \geq 1$ den folgenden Pfad $(v_1, v_2, \dots)$ in $\mathcal{T}$ mit $v_1 = r$ und $(v_i, v_{i+1}\#) \in C$ für alle $i \geq 1$:

(IA) $i = 1$. Da C eine exakte Überdeckung ist, muss ein Knoten $x \in V$ existieren mit o.B.d.A. $(r, x) \in C$ (für umgekehrte Kanten ist dies symmetrisch). Weil $r\# \notin V$ gilt, muss nach der Definition von E ein Knoten $v_2 \in N$ existieren mit $x = v_2\#$ und $(r, v_2) \in P$.

(IS) $i > 1$. Da nach (IV) $(v_{i-1}, v_i\#) \in C$ und damit $(v_i, v_i\#) \notin C$ gilt, muss also ein Knoten $x \in V \setminus \{v_i\#\}$ existieren mit o.B.d.A. $(v_i, x) \in C$ (für umgekehrte Kanten ist dies symmetrisch). Nach der Definition von E muss dann also ein $v_{i+1} \in N$ existieren mit $x = v_{i+1}\#$ und $(v_i, v_{i+1}) \in P$.

Damit ist $(v_1, v_2, \dots)$ also ein unendlicher Pfad in $\mathcal{T}$. q.e.d.

Behauptung 3. Der Graph $\mathcal{X}(\mathcal{T})$ ist effektiv automatisch.
Beweis. Da $\mathcal{T}$ automatisch ist, ist auch die erste Teilmenge aus der Definition von V regulär. Nach den Abschlusseigenschaften regulärer Sprachen ist zudem $N \setminus \{r\}$ effektiv regulär. Zudem ist die Klasse regulärer Sprachen unter Konkatenation abgeschlossen. Also ist auch die zweite Teilmenge aus der Definition von V effektiv regulär.

Ein endlicher Automat, der die erste Teilmenge aus der Definition von E akzeptiert, kann aus dem Automaten für P berechnet werden, indem auf dem zweiten Band abschließend noch das Symbol # gelesen wird. Die Menge der inversen Tupel ist dann nach dem Fundamentalsatz ebenfalls effektiv automatisch. Die zweite Teilmenge aus der Definition kann von einem Zweiband-Automaten akzeptiert werden, dessen Bänder zunächst synchron laufen und abschließend genau ein Band noch das Symbol # liest und das entsprechend andere nur das Leerzeichen $\diamond$. Damit ist E also auch effektiv automatisch. q.e.d.

Es gilt also: $\mathcal{T}$ besitzt genau dann einen unendlichen Pfad, wenn $\mathcal{X}(\mathcal{T}) \in$ Exact Cover$^{\text{aut}}$. Damit ist Exact Cover$^{\text{aut}}$ ebenfalls Σ^1_1-hart und daher -vollständig. ∎

Kapitel 5

Erfüllbarkeitsprobleme

Ein in der Logik wichtiges Problem ist die Frage nach der Erfüllbarkeit einer aussagenlogischen Formel. Im folgenden Abschnitt soll die Erfüllbarkeit *unendlicher* aussagenlogischer Formeln überprüft werden. Hierbei soll es genügen, Formeln in Konjunktiver Normalform (kurz: KNF) zu betrachten. Da aussagenlogische Formeln jedoch nur im Endlichen definiert sind, ist zunächst die folgende Definition notwendig:

Definition 5.1. Ein **Literal** ist eine atomare Formel (dies ist ein *positives* Literal) oder die Negation einer atomaren Formel (dies ist ein *negatives* Literal). Eine **Klausel** ist eine (potentiell unendliche) Menge von Literalen. Eine **Formel in Konjunktiver Normalform** ist eine (potentiell unendliche) Menge von Klauseln.

Eine **Belegung** $\mathcal{B}$ einer Formel in Konjunktiver Normalform über der Menge A von atomaren Formeln ist eine Abbildung $\mathcal{B}\colon A \to \{0,1\}$. Eine Formel in Konjunktiver Normalform ϕ über der Menge A von atomaren Formeln heißt **erfüllbar**, wenn eine Belegung $\mathcal{B}\colon A \to \{0,1\}$ existiert, sodass für jede Klausel $C \in \phi$ ein positives Literal $a \in C$ mit $\mathcal{B}(a) = 1$ oder ein negatives Literal $\neg a \in C$ mit $\mathcal{B}(a) = 0$ existiert. Andernfalls heißt ϕ **unerfüllbar**.

Intuitiv entspricht eine solche Formel ϕ damit einer (unendlichen) Konjunktion von (unendlichen) Disjunktionen von Literalen. Dies ist also eine natürliche Erweiterung von aussagenlogischen KNF-Formeln durch unendliche Konjunktionen und Disjunktionen.

Damit kann nun die Entscheidbarkeit des Erfüllbarkeitsproblems von aussagenlogischen Formeln in Konjunktiver Normalform analysiert werden. Hierzu soll dieses Problem wie folgt als Graphproblem modelliert werden:

Eingabe: Ein rekursiver, bipartiter Graph $\mathcal{G} = (A, C, P, N)$ mit $P, N \subseteq A \times C$.
Frage: Gibt es eine Menge $S \subseteq A$, sodass für alle $c \in C$ ein $a \in S$ existiert mit $(a, c) \in P$ oder ein $a \in A \setminus S$ existiert mit $(a, c) \in N$?

Im Weiteren wird dieses Problem kurz Sat$^\infty$ genannt. Die Eingabe ist hierbei wie folgt zu interpretieren:

- A ist die Menge der atomaren Formeln.
- C ist die Menge von Repräsentanten für die Klauseln.

- Es gelte $(a, c) \in P$ (resp. $(a, c) \in N$) genau dann, wenn a ein positives (resp. negatives) Literal in den von c repräsentierten Klauseln ist.

Es soll zunächst gezeigt werden, dass die Probleme SAT^∞ und das Erfüllbarkeitsproblem von KNF-Formeln tatsächlich miteinander korrespondieren. Sei dazu $\mathcal{G} = (A, C, P, N)$ ein bipartiter Graph mit $P, N \subseteq A \times C$. Dann sei $\mathcal{F}(\mathcal{G})$ die folgende aus $\mathcal{G}$ induzierte aussagenlogische Formel:

$$\mathcal{F}(\mathcal{G}) := \Big\{ \{a \,|\, (a, c) \in P\} \cup \{\neg a \,|\, (a, c) \in N\} \,\Big|\, c \in C \Big\}.$$

Die Gültigkeit der folgenden Aussage ist damit leicht zu sehen und wird deshalb hier nicht vollständig bewiesen.

Lemma 5.2. *Sei $\mathcal{G} = (A, C, P, N)$ ein bipartiter Graph mit $P, N \subseteq A \times C$. Dann gilt:*

$$\mathcal{G} \in \text{SAT}^\infty \Leftrightarrow \mathcal{F}(\mathcal{G}) \text{ ist erfüllbar.}$$

Beweisidee. „$\Rightarrow$“: Sei $S \subseteq A$ eine Menge, die die Bedingungen aus SAT^∞ für $\mathcal{G}$ erfüllt. Eine $\mathcal{F}(\mathcal{G})$ erfüllende Belegung ist dann:

$$\mathcal{B} \colon A \to \{0, 1\} \colon a \mapsto \begin{cases} 1 & \text{, falls } a \in S \\ 0 & \text{, falls } a \in A \setminus S. \end{cases}$$

„$\Leftarrow$“: Sei $\mathcal{B} \colon A \to \{0, 1\}$ eine $\mathcal{F}(\mathcal{G})$ erfüllende Belegung. Dann ist

$$S := \{a \in A \,|\, \mathcal{B}(a) = 1\} \subseteq A$$

eine Menge, die die Bedingungen aus SAT^∞ für $\mathcal{G}$ erfüllt. ∎

Hirst und Harel haben in [HH96, Proposition 14] gezeigt, dass SAT^∞ Σ^1_1-vollständig ist. Es soll stattdessen nun die folgende Einschränkung dieses Problems betrachtet werden:

Eingabe: Ein *automatischer*, bipartiter Graph $\mathcal{G} = (A, C, P, N)$ mit $P, N \subseteq A \times C$.
Frage: Gibt es eine Menge $S \subseteq A$, sodass für alle $c \in C$ ein $a \in S$ existiert mit $(a, c) \in P$ oder ein $a \in A \setminus S$ existiert mit $(a, c) \in N$?

Dieses Problem wird im Weiteren als SAT^{aut} bezeichnet. Da SAT^{aut} eine Teilmenge von SAT^∞ ist, gilt auch Lemma 5.2 analog für SAT^{aut}.

5.1 Der unbeschränkte Fall

Zunächst soll das allgemeine SAT^{aut}-Problem untersucht werden. Für dieses wird im Folgenden die Σ^1_1-Vollständigkeit gezeigt. In den weiteren Abschnitten werden

noch einige Einschränkungen von Sat^{aut} betrachtet, sodass sich diese in der arithmetischen Hierarchie wiederfinden.

Der Beweis zum folgenden Satz ist eine Variation des Beweises der Σ_1^1-Vollständigkeit von Sat^{∞} aus [HH96, Proposition 14].

Satz 5.3. *Das Problem* Sat^{aut} *ist* Σ_1^1*-vollständig.*

Beweis. Sei $\mathcal{G} = (A, C, P, N)$ wie in Sat^{aut} beschrieben. Da die Mengen A, C, P und N auch rekursiv sind und Sat^{∞} in Σ_1^1 ist, ist auch Sat^{aut} in Σ_1^1. Es bleibt also noch die Σ_1^1-Härte von Sat^{aut} zu zeigen. Hierzu soll $\text{Exact Cover}^{\text{aut}}$, welches nach Satz 4.2 Σ_1^1-hart ist, auf Sat^{aut} reduziert werden.

Idee. Kodiere den automatischen Graphen $\mathcal{G} = (V, E)$ als einen Graphen $\mathcal{S}(\mathcal{G}) = (A, C, P, N)$ wie folgt: Jede ungerichtete Kante $(x, y) \in E$ wird einer Atomformel zugeordnet. Eine Menge $S \subseteq A$ mit den Eigenschaften aus Sat^{aut} enthält dann die Kante $(x, y) \in E$ genau dann, wenn (x, y) oder (y, x) in einer exakten Überdeckung von $\mathcal{G}$ enthalten ist. Dies wird wie folgt realisiert: Jedem Knoten $v \in V$ wird eine Klausel zugeordnet, die alle Kanten (d.h. Literale) positiv enthält, in denen v vorkommt. Damit wird sichergestellt, dass jedem Knoten mindestens eine Kante zugeordnet wird. Zudem wird für jedes Paar von Kanten mit einem gemeinsamen Knoten (d.h. $u, v, w \in V$ paarweise verschieden mit $(u, v), (u, w) \in E$) eine Klausel zugeordnet, mit der die Formel $(u, v) \to \neg(u, w)$ realisiert wird. Damit wird schließlich sichergestellt, dass zu jedem Knoten höchstens eine Kante in der Lösungsmenge S enthalten ist. Es wird also die folgende Formel konstruiert:

$$\bigwedge_{v \in V} \bigvee_{\substack{u \in V: \\ (u,v) \in E}} (u, v) \wedge \bigwedge_{\substack{u,v \in V: \\ (u,v) \in E}} (u, v) \leftrightarrow (v, u)$$
$$\wedge \bigwedge_{\substack{u,v,w \in V, v \neq w: \\ (u,v),(u,w) \in E}} (u, v) \to \neg(u, w)$$

Sei also $\mathcal{G} = (V, E)$ ein automatischer, ungerichteter Graph. Konstruiere nun hieraus einen Graphen $\mathcal{S}(\mathcal{G}) = (A, C, P, N)$ mit den folgenden Eigenschaften:

- $A := E$,
- $C := V \cup E \cup \{(x, y, z) \mid (x, y), (x, z) \in E, y \neq z\}$,
- $P := \{((x, y), x), ((y, x), x) \mid (x, y) \in E\} \cup \{((y, x), (x, y)) \mid (x, y) \in E\}$,
- $N := \{((x, y), (x, y, z)), ((x, z), (x, y, z)) \mid (x, y), (x, z) \in E, y \neq z\}$
  $\cup \{((x, y), (x, y)) \mid (x, y) \in E\}$.

Anmerkung. Der Graph $\mathcal{S}(\mathcal{G})$ wird als bipartit angenommen, d.h. es handelt sich um disjunkte Kopien von E, die in den Mengen A und C enthalten sind.

Behauptung 1. Hat $\mathcal{G}$ eine exakte Überdeckung, so existiert eine Menge $S \subseteq A$, sodass für alle $c \in C$ ein $a \in S$ existiert mit $(a, c) \in P$ oder ein $a \in A \setminus S$ existiert mit $(a, c) \in N$.
Beweis. Sei $X \subseteq E$ eine exakte Überdeckung von $\mathcal{G}$. Dann definiere

$$S := X \cup X^{-1} \subseteq A.$$

Sei nun $c \in C$ eine beliebige Klausel aus $\mathcal{S}(\mathcal{G})$. Dazu betrachte die folgende Fallunterscheidung:

(Fall 1) Es existiert ein $x \in V$ mit $c = x$.

Da X eine exakte Überdeckung ist, gibt es ein $y \in V$ mit $(x, y) \in X$ oder $(y, x) \in X$. Es gilt dann $(x, y) \in S$. Nach Definition von P gilt zudem $((x, y), c) \in P$.

(Fall 2) Es existieren $x, y \in V$ mit $(x, y) = c$.

(2.1) Es gilt $(x, y), (y, x) \notin X$.
Dann gilt $(x, y) \notin S$. Nach der Definition von N gilt aber auch $((x, y), c) \in N$.

(2.2) Es gilt $(x, y) \in X$ oder $(y, x) \in X$.
Dann gilt $(y, x) \in S$. Nach der Definition von P gilt aber auch $((y, x), c) \in P$.

(Fall 3) Es existieren $x, y, z \in V$ mit $(x, y, z) = c$.

Es gilt also $(x, y), (x, z) \in E$ und $y \neq z$.

(3.1) Es gilt $(x, y), (y, x) \notin X$.
Dann gilt $(x, y) \notin S$. Nach Definition von N gilt aber auch $((x, y), c) \in N$.

(3.2) Es gilt $(x, y) \in X$ oder $(y, x) \in X$.
Dann gilt $(x, y) \in S$. Da X exakte Überdeckung ist gilt zudem $(x, z), (z, x) \notin X$, d.h. $(x, z) \notin S$. Nach der Definition von N gilt aber auch $((x, z), c) \in N$.

Da $c \in C$ beliebig war, gilt dies also für alle Klauseln. Somit hat S die gewünschte Form. q.e.d.

Behauptung 2. Existiert eine Menge $S \subseteq A$, sodass für alle $c \in C$ ein $a \in S$ existiert mit $(a, c) \in P$ oder ein $a \in A \setminus S$ existiert mit $(a, c) \in N$, so hat $\mathcal{G}$ eine exakte Überdeckung.
Beweis. Sei nun $S \subseteq A$ wie oben. S ist symmetrisch, denn: Sei $(x, y) \in S$. Da die Klausel $(x, y) \in C$ erfüllt wird, muss $(y, x) \in S$ oder $(x, y) \notin S$ gelten. Nach der Voraussetzung gilt also $(y, x) \in S$.

Definiere die Menge

$$X := \{(x, y) \mid (x, y) \in S, x \leq y\},$$

wobei $(V, \leq)$ eine beliebige lineare Ordnung ist. Sei $x \in V$ ein beliebiger Knoten. Da S die Klausel $x \in C$ erfüllt, gibt es mindestens ein $y \in V$ mit $(x, y) \in S$ oder $(y, x) \in S$. Da S symmetrisch ist, muss dann also $(x, y), (y, x) \in S$ gelten. Sei nun $z \in V$ mit $z \neq y$ und $(x, z) \in E$. Da S die Klausel $(x, y, z) \in E$ erfüllt und $(x, y) \in S$ negativ in dieseer Klausel vorkommt, muss $(x, z) \notin S$ gelten. Es existiert also genau ein $y \in V$ mit $(x, y) \in S$. Nach der Definition von X gibt es dann für jedes $x \in V$ auch nur genau ein $y \in V$ mit $(x, y) \in X$ oder $(y, x) \in X$. Damit ist X also eine exakte Überdeckung von $\mathcal{G}$. q.e.d.

Aus den Behauptungen 1 und 2 folgt, dass $\mathcal{G}$ genau dann eine exakte Überdeckung besitzt, wenn in $\mathcal{S}(\mathcal{G})$ eine Menge $S \subseteq A$ mit den Eigenschaften aus SAT^{aut} existiert. Es bleibt nun noch zu zeigen, dass $\mathcal{S}(\mathcal{G})$ effektiv automatisch ist.

Behauptung 3. Der Graph $\mathcal{S}(\mathcal{G})$ ist effektiv automatisch.
Beweis. Die Mengen A und C sind per Definition disjunkt. Da in $\mathcal{S}(\mathcal{G})$ beide Mengen die Relation E enthalten, wird die Kopie von E in C als $\{\#\}E$ kodiert, wobei $\# \notin \Sigma$ gilt, falls Σ das Alphabet der automatischen Darstellung von $\mathcal{G}$ ist.

Die Menge A sowie die erste Teilmenge von C sind automatisch, da $\mathcal{G}$ ein automatischer Graph ist. Da die Klasse regulärer Sprachen unter Konkatenation effektiv abgeschlossen ist, ist die zweite Teilmenge von C automatisch. Die dritte Teilmenge von C ist nach dem Fundamentalsatz effektiv automatisch. Nach den Abschlusseigenschaften ist also C automatisch.

Die Mengen P und N sind ebenfalls nach dem Fundamentalsatz effektiv automatisch und damit auch $\mathcal{S}(\mathcal{G})$. q.e.d.

Damit gilt also: $\mathcal{G} \in \text{Exact Cover}^{\text{aut}}$ genau dann, wenn $\mathcal{S}(\mathcal{G}) \in \text{SAT}^{\text{aut}}$. Also ist SAT^{aut} Σ^1_1-hart und damit -vollständig. ∎

Bemerkung 5.4. Analog zu SAT^{aut} kann auch untersucht werden, ob $\mathcal{F}(\mathcal{G})$ eines Graphen $\mathcal{G} = (A, C, P, N)$ eine Tautologie ist. Dieses im Folgenden TAUT^{aut} genannte Problem ist wie folgt definiert:

> **Eingabe:** Ein automatischer Graph $\mathcal{G} = (A, C, P, N)$ wie in SAT^{aut}.
> **Frage:** Besitzt *jede* Menge $S \subseteq A$ die Eigenschaft, dass für alle $c \in C$ ein $a \in S$ existiert mit $(a, c) \in P$ oder ein $a \in A \setminus S$ existiert mit $(a, c) \in N$?

Behauptung. Das Problem TAUT^{aut} ist entscheidbar.
Beweis. Sei $\mathcal{G} = (A, C, P, N)$ wie oben angegeben. Die Formel $\mathcal{F}(\mathcal{G})$ ist genau dann eine Tautologie, wenn alle ihre Klauseln Tautologien sind. Dies ist der Fall, wenn jede Klausel sowohl ein Literal L als auch dessen Negation $\overline{L}$ enthält.
Betrachte die folgende Formel:

$$\phi(c) := \exists a \colon (C(c) \wedge A(a) \wedge P(a, c) \wedge N(a, c)).$$

Dann ist $\phi^{\mathcal{G}}$ die Menge aller tautologischen Klauseln. Nach dem Fundamentalsatz ist diese Menge effektiv automatisch. Es kann also entschieden werden, ob $\phi^{\mathcal{G}} = C$ gilt. q.e.d.

5.2 Der endliche Fall

Es soll nun der deutlich natürlichere Fall betrachtet werden, in dem jeder Klausel nur endlich viele Literale zugeordnet werden. Für diese Art von Formeln gilt sogar der Endlichkeitssatz der Aussagenlogik, denn: Sei $\mathcal{G}$ ein Graph mit

$$\mathcal{F}(\mathcal{G}) = \{C_i \,|\, i \geq 0\}$$

und $|C_i| < \infty$ für alle $i \geq 0$. Dann ist $\mathcal{F}(\mathcal{G})$ eine (unendliche) Menge von endlichen Disjunktionen, d.h. aussagenlogischen Formeln. Dies ist bekanntlich die Prämisse für den Endlichkeitssatz.

Es wird im Folgenden gezeigt, dass das Problem SAT^{aut} mit ausschließlich endlichen Klauseln Π_1^0-vollständig ist.

Proposition 5.5. *Das Problem* SAT^{aut} *mit der Einschränkung, dass jeder Klausel nur endlich viele Literale zugeordnet sind, ist in* Π_1^0.

Beweis. Betrachte den folgenden Algorithmus:

Eingabe: $\mathcal{G} = (A, C, P, N)$ wie oben beschrieben

(1) Sei $C_1, C_2, \ldots$ die längenlexikografische Aufzählung von C.
(2) $n \leftarrow 0$
(3) $K \leftarrow \emptyset$
(4) **while** *Klauselmenge K ist aussagenlogisch erfüllbar* **do**
(5) $\quad n \leftarrow n + 1$
(6) $\quad K \leftarrow K \cup \{\{a \,|\, (a, C_n) \in P\} \cup \{\neg a \,|\, (a, C_n) \in N\}\}$
(7) **end while**
(8) **return** *„unerfüllbar“*

Anmerkung. Die längenlexikografische Aufzählung von C in Zeile 1 ist effektiv automatisch. Die Menge $\{a \,|\, (a, C_n) \in P\} \cup \{\neg a \,|\, (a, C_n) \in N\}$ in Zeile 6 ist nach dem Fundamentalsatz ebenfalls effektiv automatisch und nach der Voraussetzung endlich.

Es muss nun noch die Korrektheit des Algorithmus gezeigt werden.

Behauptung 1. Gibt der Algorithmus „unerfüllbar“ aus, dann gilt auch $\mathcal{G} = (A, C, P, N) \notin \text{SAT}^{\text{aut}}$.
Beweis. Der Algorithmus führt im n-ten Durchlauf der While-Schleife einen Erfüllbarkeitsalgorithmus (z.B. Resolution) auf der Formelmenge

$$\mathcal{F}_n(\mathcal{G}) := \Big\{\{a \,|\, (a, C_i) \in P\} \cup \{\neg a \,|\, (a, C_i) \in N\} \,\Big|\, 1 \leq i \leq n\Big\} \subseteq \mathcal{F}(\mathcal{G})$$

aus, d.h. auf den längenlexikografischen ersten n Klauseln.

Da der Algorithmus „unerfüllbar“ ausgegeben hat, muss also eine Zahl $n \geq 1$ existieren, sodass die Formelmenge $\mathcal{F}_n(\mathcal{G})$ unerfüllbar ist. Da $\mathcal{F}_n(\mathcal{G})$ jedoch eine Teilmenge von $\mathcal{F}(\mathcal{G})$ ist, muss daher auch $\mathcal{F}(\mathcal{G})$ unerfüllbar sein. Nach Lemma 5.2 gilt dann $\mathcal{G} \notin \text{SAT}^{\text{aut}}$. q.e.d.

Behauptung 2. Gilt $\mathcal{G} = (A, C, P, N) \notin \text{SAT}^{\text{aut}}$, so gibt der Algorithmus „unerfüllbar“ aus.
Beweis. Da $\mathcal{G} = (A, C, P, N) \notin \text{SAT}^{\text{aut}}$, gilt nach Lemma 5.2 auch: $\mathcal{F}(\mathcal{G})$ ist unerfüllbar. Nach dem Endlichkeitssatz der Aussagenlogik gibt es also eine unerfüllbare endliche Teilformel von $\mathcal{F}(\mathcal{G})$. Seien $C_{i_1}, C_{i_2}, \ldots, C_{i_m}$ die Klauseln in dieser Teilformel mit o.B.d.A. $i_1 < i_2 < \ldots < i_m$. Die Formel $\bigwedge_{k=1}^{m} C_{i_k}$ ist unerfüllbar, also auch $\mathcal{F}_{i_m}(\mathcal{G}) = \bigwedge_{l=1}^{i_m} C_l$. Daher terminiert der Algorithmus spätestens nach dem i_m-ten Schleifendurchlauf mit der Ausgabe „unerfüllbar“. q.e.d.

Damit wurde die Korrektheit des Algorithmus gezeigt. ∎

Bemerkung 5.6. Betrachte das Problem Exact Cover$^{\text{aut}}$ mit der Einschränkung, dass jeder Knoten mit höchstens endlich vielen weiteren Knoten verbunden ist. In dem im Beweis von Satz 5.3 konstruierten Graphen $\mathcal{S}(\mathcal{G})$ werden in diesem Fall jeder Klausel nur endlich viele Literale zugeordnet. Es kann also gefolgert werden, dass dieses Problem nach Proposition 5.5 ebenfalls in Π_1^0 ist.

Es soll nun noch gezeigt werden, dass Π_1^0 auch eine untere Schranke in der arithmetischen Hierarchie für das Problem SAT$^{\text{aut}}$ mit ausschließlich endlichen Klauseln ist. Hierbei wird sogar gezeigt, dass SAT$^{\text{aut}}$ mit der Einschränkung, dass allen Klauseln höchstens *zwei* Literale zugeordnet werden, Π_1^0-hart ist. Dazu wird das Komplement des als *Post'sches Korrespondenzproblem* (kurz: PCP) bekannten folgenden Problems verwendet:

Eingabe: Eine endliche Folge von Wortpaaren $I = ((x_1, y_1), \ldots, (x_n, y_n))$ mit $n \geq 1$ über dem Alphabet Σ.
Frage: Existiert eine Folge $(i_1, \ldots, i_m)$ mit $1 \leq i_k \leq n$ für alle $1 \leq k \leq m$, sodass $x_{i_1} x_{i_2} \ldots x_{i_m} = y_{i_1} y_{i_2} \ldots y_{i_m}$?

Nach [Pos46] ist PCP Σ_1^0-vollständig.

Proposition 5.7. *Das Problem* SAT$^{\text{aut}}$ *mit der Einschränkung, dass jeder Klausel nur* maximal zwei *Literale zugeordnet werden, ist Π_1^0-hart.*

Beweis. Mithilfe des Komplements von PCP soll nun die Π_1^0-Härte von SAT$^{\text{aut}}$ mit Klauseln mit höchstens zwei Literalen gezeigt werden.

Idee. Aus der Instanz $I = ((x_1, y_1), \ldots, (x_n, y_n))$ über Σ wird wie folgt ein Graph $\mathcal{K}(I)$ mit den Eigenschaften aus SAT$^{\text{aut}}$ konstruiert: Die atomaren Formeln sind Wortpaare $(u, v) \in (\Sigma^*)^2$. $(\varepsilon, \varepsilon)$ darf nur mit 1 und (u, u) mit $u \in \Sigma^+$ darf nur mit 0 belegt werden. Zudem wird für jedes Wortpaar (u, v) und jedes $1 \leq i \leq n$ eine

Klausel $(u, v) \to (ux_i, vy_i)$ eingefügt. Es wird also die folgende Formel realisiert:

$$(\varepsilon, \varepsilon) \wedge \bigwedge_{u \in \Sigma^+} \neg(u, u) \wedge \bigwedge_{i=1}^{n} \bigwedge_{u,v \in \Sigma^*} (u, v) \to (ux_i, vy_i).$$

Induktiv ergibt sich dann die Belegung 1 für jedes $(x_{i_1} \dots, x_{i_m}, y_{i_1} \dots y_{i_m})$ und damit ein Widerspruch, falls I eine PCP-Lösung besitzt.

Sei also $I = ((x_1, y_1), \dots, (x_n, y_n))$ mit $n \geq 1$ eine endliche Folge von Wortpaaren über dem Alphabet Σ. Konstruiere nun den folgenden bipartiten Graphen $\mathcal{K}(I) = (A, P, C, N)$ mit den Eigenschaften:

- $A := (\Sigma^*)^2$,
- $C := \Sigma^* \cup (\{1, \dots, n\} \times (\Sigma^*)^2)$,
- $P := \{((\varepsilon, \varepsilon), \varepsilon)\} \cup \{((ux_i, vy_i), (i, u, v)) \,|\, u, v \in \Sigma^*, 1 \leq i \leq n\}$,
- $N := \{((u, u), u) \,|\, u \in \Sigma^+\} \cup \{((u, v), (i, u, v)) \,|\, u, v \in \Sigma^*, 1 \leq i \leq n\}$.

Behauptung 1. Gilt $I \notin$ PCP, so existiert eine Menge $S \subseteq A$, sodass für alle $c \in C$ ein $a \in S$ existiert mit $(a, c) \in P$ oder ein $a \in A \setminus S$ existiert mit $(a, c) \in N$.
Beweis. Sei also $I \notin$ PCP. Definiere nun induktiv über $j \geq 0$ die Mengen $S_j \subseteq A$ wie folgt:

(IA) $j = 0$. $S_0 := \{(\varepsilon, \varepsilon)\} \subseteq A$.

(IS) $j > 0$. $S_j := \{(ux_i, vy_i) \,|\, (u, v) \in S_{j-1}, 1 \leq i \leq n\} \subseteq A$.

Dann sei $S := \bigcup_{j \geq 0} S_j \subseteq A$ die zu untersuchende Menge in $\mathcal{K}(I)$.

Sei nun also $c \in C$ eine beliebige Klausel. Dann tritt einer der folgenden drei Fälle ein:

(Fall 1) $c = \varepsilon$.

Es gilt nach Definition $(\varepsilon, \varepsilon) \in S_0 \subseteq S$. Nach der Definition von P gilt zudem $((\varepsilon, \varepsilon), \varepsilon) \in P$.

(Fall 2) $c \in \Sigma^+$.

Angenommen, es gilt $(c, c) \in S$. Dann existiert ein $m > 0$ mit $(c, c) \in S_m$. Es ergibt sich dann nach der Definition von S_m induktiv eine Folge von Indizes $(i_1, \dots, i_m)$ mit $1 \leq i_1, \dots, i_m \leq m$ und $x_{i_1} \dots x_{i_m} = c = y_{i_1} \dots y_{i_m}$ (es gilt $(x_{i_1} \dots x_{i_j}, y_{i_1} \dots y_{i_j}) \in S_j$ für alle $0 \leq j \leq m$). Diese Folge ist jedoch eine Lösung des PCP in der Instanz I. Dies ist ein Widerspruch zur Voraussetzung $I \notin$ PCP. Damit gilt aso $(c, c) \in A \setminus S$. Zudem gilt $((c, c), c) \in N$ nahc der Definition von N.

(Fall 3) $c = (i, u, v)$.

(3.1) Es gilt $(u, v) \in A \setminus S$.

Nach der Definition von N gilt $((u, v), (i, u, v)) \in N$.

(3.2) Es gilt $(u, v) \in S$.

Es existiert also ein $j \geq 0$ mit $(u, v) \in S_j$. Nach der Definition von S gilt dann aber auch $(u, x_i, vy_i) \in S_{j+1} \subseteq S$. Nach der Definition von P gilt zudem $((ux_i, vy_i), (i, u, v)) \in P$.

Somit erfüllt $S \subseteq A$ also die gewünschten Bedingungen aus SAT^{aut}. q.e.d.

Behauptung 2. Existiert eine Menge $S \subseteq A$, sodass für alle $c \in C$ ein $a \in S$ existiert mit $(a, c) \in P$ oder ein $a \in A \setminus S$ existiert mit $(a, c) \in N$, so gilt auch $I \notin \text{PCP}$.
Beweis. Sei also $S \subseteq A$ eine Menge, die die Bedingungen aus SAT^{aut} in $\mathcal{K}(I)$ erfüllt. Es soll zunächst per Induktion über $m \geq 0$ gezeigt werden, dass für jede endliche Indexfolge $(i_1, \ldots, i_m)$ mit $1 \leq i_1, \ldots, i_m \leq n$ gilt:

$$(x_{i_1} \ldots x_{i_m}, y_{i_1} \ldots y_{i_m}) \in S.$$

(IA) $m = 0$. Da S die Klausel $\varepsilon \in C$ erfüllt, muss $(\varepsilon, \varepsilon) \in S$ gelten.

(IV) $m > 0$. Es gilt für alle Folgen $(i_1, \ldots, i_m)$ mit $1 \leq i_1, \ldots, i_m \leq n$: $(x_{i_1} \ldots x_{i_m}, y_{i_1} \ldots y_{i_m}) \in S$.

(IS) $m > 0$. Sei also $(i_1, \ldots, i_m)$ eine beliebige, aber feste Indexfolge mit $1 \leq i_1, \ldots, i_m \leq n$. Sei zudem $1 \leq i_{m+1} \leq n$ beliebig. Betrachte nun die Klausel $(i_{m+1}, x_{i_1} \ldots x_{i_m}, y_{i_1} \ldots y_{i_m}) \in C$. Diese enthält einerseits das Literal $(x_{i_1} \ldots x_{i_m}, y_{i_1} \ldots y_{i_m}) \in A$ negativ und andererseits das Literal $(x_{i_1} \ldots x_{i_m} x_{i_{m+1}}, y_{i_1} \ldots y_{i_m} y_{i_{m+1}}) \in A$ positiv. Nach der (IV) gilt $(x_{i_1} \ldots x_{i_m}, y_{i_1} \ldots y_{i_m}) \in S$. Da aber S die Bedingungen aus SAT^{aut} erfüllt, muss also auch $(x_{i_1} \ldots x_{i_m} x_{i_{m+1}}, y_{i_1} \ldots y_{i_m} y_{i_{m+1}}) \in S$ gelten.

Da die Indizes $1 \leq i_1, \ldots, i_m, i_{m+1} \leq n$ beliebig waren, gilt dies also für alle Indexfolgen $(i_1, \ldots, i_m, i_{m+1})$.

Angenommen, es gäbe nun eine PCP-Lösung $(i_1, \ldots, i_m)$ mit $m > 0$ und $1 \leq i_1, \ldots, i_m \leq n$, d.h. es gilt $x_{i_1} \ldots x_{i_m} = y_{i_1} \ldots y_{i_m}$. Es gibt also auch eine Klausel $x_{i_1} \ldots x_{i_m} \in C$, die ausschließlich $(x_{i_1}, \ldots, x_{i_m}, y_{i_1} \ldots y_{i_m}) \in A$ negativ enthält. Da S die Bedingungen aus SAT^{aut} erfüllt, muss also $(x_{i_1}, \ldots, x_{i_m}, y_{i_1} \ldots y_{i_m}) \in S \setminus A$ gelten. Es wurde jedoch eben gezeigt, dass $(x_{i_1}, \ldots, x_{i_m}, y_{i_1} \ldots y_{i_m}) \in S$ gilt. Dies ist ein Widerspruch.

Somit kann es also keine Indexfolge $(i_1, \ldots, i_m)$ in der Instanz I mit $m > 0$, $1 \leq i_1, \ldots, i_m \leq n$ und $x_{i_1} \ldots x_{i_m} = y_{i_1} \ldots y_{i_m}$ geben. Es gilt also $I \notin \text{PCP}$. q.e.d.

Aus den Behauptungen 1 und 2 ergibt sich zunächst der folgende Zusammenhang: Es gilt genau dann $I \notin \text{PCP}$, wenn in $\mathcal{K}(I)$ eine Menge $S \subseteq A$ mit den Eigenschaften aus SAT^{aut} existiert. Es muss nur noch gezeigt werden, dass $\mathcal{K}(I)$ aus der Instanz I berechnet werden kann und automatisch ist.

Behauptung 3. Der Graph $\mathcal{K}(I)$ ist effektiv automatisch.
Beweis. Die Menge A ist offensichtlich effektiv automatisch (verwende einen Zweiband-Automaten, der auf jedem Band alle Wörter über dem Alphabet Σ akzeptiert), genauso wie die Menge C (verwende für die zweite Teilmenge einen Dreiband-Automaten, welcher wie der für A funktioniert, aber auf dem ersten Band nur ein Symbol der Menge $\{1, \ldots, n\}$ akzeptiert).

Für jedes $1 \leq i \leq n$ kann ein Fünfband-Automat berechnet werden, der auf den Bändern 4 und 5 wie der Automat von A funktioniert. Auf Band 3 wird nur das Symbol i gelesen. Und die Bänder 1 und 2 lesen zunächst dieselben Wörter wie die Bänder 4 bzw. 5 und lesen abschließend noch mit x_i bzw. y_i jeweils ein endliches Wort. Die Menge P ist dann die Vereinigung der akzeptierten Sprachen dieser Automaten (plus eines Automaten, der auf seinen drei Bändern nur das leere Wort akzeptiert), welche nach den Abschlusseigenschaften automatischer Relationen effektiv automatisch ist.

Die erste Teilmenge von N kann durch einen Dreiband-Automaten akzeptiert werden, dessen Bänder allesamt synchron laufen. Die zweite Teilmenge dagegen von einem Fünfband-Automaten, der ähnlich funktioniert wie der für die Menge P (ohne die Wörter x_i und y_i am Ende zu lesen). Auch damit ist diese Menge effektiv automatisch nach den Abschlusseigenschaften. q.e.d.

Es folgt also: $I \notin \text{PCP}$ genau dann, wenn $\mathcal{K}(I) \in \text{SAT}^{\text{aut}}$. Da das Komplement von PCP nach [Pos46] Π_1^0-hart ist, ist auch SAT^{aut} Π_1^0-hart. ∎

Aus den Propositionen 5.5 und 5.7 folgt schließlich unmittelbar:

Satz 5.8. *Das Problem* SAT^{aut} *mit der Einschränkung, dass jeder Klausel nur endlich viele Literale zugeordnet sind, ist* Π_1^0*-vollständig.*

Bemerkung 5.9. Nachdem gezeigt wurde, dass SAT^{aut} mit der Einschränkung, dass jeder Klausel höchstens zwei Literale zugeordnet werden, Π_1^0-vollständig ist, soll noch das Problem SAT^{aut} mit der Einschränkung, dass jeder Klausel *höchstens ein* Literal zugeordnet wird, betrachtet werden. Hierfür gibt sich sogar die Entscheidbarkeit.
Beweis. Sei also $\mathcal{G} = (A, C, P, N)$ wie in SAT^{aut} mit höchstens einem Literal pro Klausel. Dann ist $\mathcal{F}(\mathcal{G})$ genau dann erfüllbar, wenn es keine leere Klausel gibt und es keine atomare Formel $a \in A$ gibt, die sowohl positiv als auch negativ in Literalen vorkommt. Beide Bedingungen lassen sich als prädikatenlogische Formeln erster Stufe ausdrücken und können nach dem Fundamentalsatz entschieden werden. q.e.d.

Nun stellt sich jedoch noch die Frage, in welcher Klasse der arithmetischen Hierarchie SAT^{∞} mit ausschließlich endlichen Klauseln einzuordnen ist. Der Algorithmus aus dem Beweis von Proposition 5.5 kann für dieses Problem nicht angewendet werden, da die in Zeile 6 angegebene Menge für rekursive Strukturen im Allgemeinen nicht berechnet werden kann.

Sei also $\mathcal{G} = (A, C, P, N)$ wie in Sat^∞ mit der gewünschten Einschränkung. Dann ist $\mathcal{G} \in \text{Sat}^\infty$ genau dann, wenn gilt: Für jede *endliche* Klauselmenge $X \subseteq C$ gibt es *endliche*, disjunkte Mengen $S_+, S_- \subseteq A$, sodass gilt:

$$\bigwedge_{c \in X} \left(\bigvee_{a \in S_+} P(a, c) \vee \bigvee_{a \in S_-} N(a, c) \right).$$

Die Mengen $S_+, S_- \subseteq A$ repräsentieren die Lösungsmenge $S \subseteq A$ sowie $A \setminus S$ in der jeweiligen endlichen Formelmenge. Es ist zudem entscheidbar, ob die oben genannte Formel gilt. Damit ist gezeigt, dass Sat^∞ mit endlichen Klauseln in Π_2^0 enthalten ist. Es ist jedoch nicht bekannt, ob dieses Problem auch Π_2^0-hart ist.

5.3 Der beschränkt unendliche Fall

Schlussendlich soll noch eine weitere, weniger restriktive Einschränkung von Sat^{aut} betrachtet werden: Die Formel $\mathcal{F}(\mathcal{G})$ besitzt höchstens endlich viele unendliche Klauseln.

Der Algorithmus aus dem Beweis von Proposition 5.5 lässt sich auf dieses Problem nicht anwenden. Beispielsweise zu der folgenden unerfüllbaren Formel gibt es einen automatischen Graphen $\mathcal{G} = (A, C, P, N)$, sodass der genannte Algorithmus nicht terminiert:

$$\mathcal{F}(\mathcal{G}) = \Big\{ \{A_i\} \,\Big|\, i \geq 0 \Big\} \cup \Big\{ \{\neg A_i \,|\, i \geq 0\} \Big\}.$$

Ursache hierfür ist: Jede endliche Klauselmenge von $\mathcal{F}(\mathcal{G})$ ist erfüllbar, jedoch $\mathcal{F}(\mathcal{G})$ nicht, d.h. der Endlichkeitssatz der Aussagenlogik gilt für diese Formel nicht. Die Abbruchbedingung der Schleife dieses Algorithmus wird also in keinem Durchlauf erfüllt. Also terminiert der Algorithmus nicht.

Es soll in diesem Abschnitt gezeigt werden, dass dieses Problem Σ_2^0-vollständig ist. Die folgende Proposition zeigt zunächst die Zugehörigkeit zur Klasse Σ_2^0.

Proposition 5.10. *Das Problem* Sat^{aut} *mit der Einschränkung, dass höchstens endlich viele Klauseln mit unendlich vielen zugeordneten Literalen existieren, ist in* Σ_2^0.

Beweis. Betrachte den folgenden Algorithmus, der von einer nichtdeterministischen Orakel-Turingmaschine mit dem Orakel Sat^{aut} unter der Einschränkung, dass jede Klausel höchstens endlich viele Literale enthält, berechnet werden kann:

Eingabe: $\mathcal{G} = (A, C, P, N)$ wie oben beschrieben
(1) $C \leftarrow \{x \mid \forall y\colon (\neg P(y,x) \vee \neg N(y,x))\}$
(2) $(P, N) \leftarrow (P \cap A \times C, N \cap A \times C)$
(3) $I \leftarrow \{x \mid \exists^\infty y\colon (P(y,x) \vee N(y,x))\}$
(4) $C_1, C_2, \ldots, C_n$ sei die längenlexikografische Aufzählung von I.
(5) **for** $i \leftarrow 1$ **to** n **do** $A_i \leftarrow \{a \mid (a, C_i) \in P \cup N\}$
(6) $t = (a_1, \ldots, a_n)$ sei nichtdeterministisch gewähltes Element aus $A_1 \times \cdots \times A_n$.
(7) **if** t *ist widersprüchlich* **then return** *„?“*
(8) $A' \leftarrow A \setminus \{a_1, \ldots, a_n\}$
(9) $C' \leftarrow \{x \in C \mid \bigwedge_{i=1}^{n}((P(a_i, C_i) \rightarrow \neg P(a_i, x)) \wedge (N(a_i, C_i) \rightarrow \neg N(a_i, x)))\}$
(10) $(P', N') \leftarrow (P \cap A' \times C', N \cap A' \times C')$
(11) **if** $(A', C', P', N') \in \text{SAT}^{\text{aut}}$ *(aus Abschnitt 5.2)* **then return** *„erfüllbar“*
(12) **else return** *„?“*

Anmerkung. In den Zeilen 1-2 werden zunächst sämtliche tautologischen Klauseln entfernt. Die Frage „$t = (a_1, \ldots, a_n)$ ist widersprüchlich“ in Zeile 7 hat die folgende Bedeutung: Es existieren zwei Indizes $1 \leq i < j \leq n$ mit $a_i = a_j$ und a_i kommt in Klausel C_i positiv und in Klausel C_j negativ (oder umgekehrt) vor. Der Algorithmus setzt die Belegungen der Literale in t zu 1 und entfernt anschließend alle Klauseln, deren Belegungen dadurch ebenfalls zu 1 gewählt werden (insbesondere sind dies die unendlichen Klauseln). Dass die einzelnen Zeilen berechenbar sind, folgt unmittelbar aus dem Fundamentalsatz, den Abschlusseigenschaften regulärer Sprachen sowie der effektiven Automatizität der längenlexikografischen Ordnung über einer beliebigen Sprache.

Es handelt sich damit also tatsächlich um einen Algorithmus mit Orakel in Zeile 11. Da der dort zu prüfende Graph nur endliche Klauseln enthält, ist diese Frage nach Proposition 5.5 in Π_1^0. Es muss nun nur noch die Korrektheit des Algorithmus nachgewiesen werden.

Behauptung 1. Besitzt der Algorithmus bei Eingabe $\mathcal{G}$ eine Berechnung mit Ausgabe „erfüllbar“, so gilt auch $\mathcal{G} \in \text{SAT}^{\text{aut}}$.
Beweis. Es gibt also ein Tupel $t = (a_1, \ldots, a_n)$, sodass der Algorithmus „erfüllbar“ ausgibt. Nach den Anmerkungen kann angenommen werden, dass $\mathcal{G}$ keine tautologischen Klauseln enthält und $a_1, \ldots, a_n$ nicht widersprüchlich sind (d.h., wenn ein Atom in t mehrfach vorkommt, so stets positiv oder stets negativ). Sei nun S' eine Menge mit den Eigenschaften aus SAT^{aut} für den Graphen (A', C', P', N') in Zeile 11. Definiere nun die folgende Menge:

$$S := S' \cup \{a_i \mid (a_i, C_i) \in P, 1 \leq i \leq n\}.$$

Dann gelten die Bedingungen aus SAT^{aut} an S weiterhin für die Klauseln aus C'. Des Weiteren gilt für die Klauseln aus $C_i \in I$:

(Fall 1) Gilt $(a_i, C_i) \in P$, so gilt per Definition $a_i \in S$.

(Fall 2) Gilt $(a_i, C_i) \notin P$. Da a_i ein in C_i vorkommendes Atom ist, muss dann also $(a_i, C_i) \in N$ gelten. Zudem gilt per Definition $a_i \in A \setminus S$.

Es müssen nun also nur noch die Klauseln $c \in C \setminus (C' \cup I)$ betrachtet werden. Damit ist c eine endliche Klausel und es existiert ein $1 \leq i \leq n$, sodass eine der folgenden Aussagen gilt:

(Fall 1) $(a_i, C_i) \in P$ und $(a_i, c) \in P$. Dann gilt (wie oben erwähnt) $a_i \in S$.

(Fall 2) $(a_i, C_i) \in N$ und $(a_i, c) \in N$. Dann gilt (wie oben erwähnt) $a_i \in A \setminus S$.

Damit hat S tatsächlich die Bedingungen aus SAT^{aut}, d.h. es gilt $\mathcal{G} \in \text{SAT}^{\text{aut}}$. q.e.d.

Behauptung 2. Gilt $\mathcal{G} \in \text{SAT}^{\text{aut}}$, so existiert eine Berechnung des Algorithmus mit Eingabe $\mathcal{G}$ und Ausgabe „erfüllbar“.
Beweis. Es kann erneut angenommen werden, dass $\mathcal{G}$ keine tautologischen Klauseln enthält. Sei S eine Menge mit den Eigenschaften aus SAT^{aut}. Betrachte nun die unendlichen Klauseln $C_1, \ldots, C_n$ in $\mathcal{G}$. Wähle für jedes $1 \leq i \leq n$ nun ein a_i wie folgt:

(Fall 1) Existiert ein $a \in S$ mit $(a, C_i) \in P$, so wähle a_i zu einem solchen a.

(Fall 2) Existiert kein $a \in S$ mit $(a, C_i) \in P$, so existiert aber ein $a \in A \setminus S$ mit $(a, C_i) \in N$ (per Definition von S). Wähle a_i nun zu einem solchen a.

Betrachte nun die Berechnung des Algorithmus, in der das Tupel $(a_1, \ldots, a_n)$ in Zeile 6 ausgewählt wird. Dieses Tupel ist nicht widersprüchlich. Es muss nun noch gezeigt werden, dass der Graph (A', C', P', N') in Zeile 11 tatsächlich in SAT^{aut} liegt. Definere dazu die Menge $S' := S \cap A'$. Betrachte nun eine beliebige Klausel $c \in C'$. Es kommt nach der Definition von C' kein a_i $(1 \leq i \leq n)$ in c mit gleichem Vorzeichen wie in C_i vor. Es muss nun also einer der folgenden Fälle gelten

(Fall 1) Es existiert ein $a \in S$ mit $(a, c) \in P$ und $a \notin \{a_1, \ldots, a_n\}$. Nach der Definition von S' gilt dann also $a \in S'$.

(Fall 2) Es existiert ein $a \in A \setminus S$ mit $(a, c) \in N$ und $a \notin \{a_1, \ldots, a_n\}$. Nach der Definition von S' gilt dann also $a \in A' \setminus S'$.

Damit hat also S' die gewünschten Eigenschaften aus SAT^{aut}, sodass $(A', C', P', N') \in \text{SAT}^{\text{aut}}$ gilt. Der Algorithmus gibt in diesem Falle „erfüllbar“ aus. q.e.d.

Damit folgt die Korrektheit des Algorithmus. Das Problem SAT^{aut} mit endlich vielen unendlichen Klauseln ist also in Σ_2^0. ∎

Es bleibt nun noch, die Σ_2^0-Härte dieses Problems zu zeigen. Dies soll erneut mithilfe des Post'schen Korrespondenzproblems geschehen, welches nun folgendermaßen abgewandelt werden soll:

Eingabe: Eine Instanz $I = ((x_1, y_1), \ldots, (x_n, y_n))$ des PCP.
Frage: Existiert eine Folge $(i_1, \ldots, i_m)$ mit $m \geq 0$ und $1 \leq i_1, \ldots, i_m \leq n$, sodass $x_1 x_{i_1} \ldots x_{i_m} = y_1 y_{i_1} \ldots y_{i_m}$ gilt? Mit anderen Worten: Ist die Folge $(1, i_1, \ldots, i_m)$ eine PCP-Lösung von I?

Dieses Problem wird im Folgenden als MPCP[3] bezeichnet und ist nach [Sch92] Σ_1^0-vollständig. Es soll nun die Reduktion vom allgemeinen Halteproblem HP auf

[3] MPCP ist eine Abkürzung für „**M**odifiziertes **PCP**“.

MPCP aus [Sch92] erneut betrachtet werden. Sei dazu $\mathcal{M} = (Q, \Sigma, \Gamma, \Delta, q_0, \square, F)$ eine deterministische Turingmaschine und $w \in \Sigma^*$ eine Eingabe von $\mathcal{M}$. Dann ist $\mathcal{I}(\mathcal{M}, w)$ die folgende PCP-Instanz über dem Alphabet $\Gamma \uplus Q \uplus \{\#, \$\}$:

- *Erstes Wortpaar.* $(x_1, y_1) = (\$, \$\square q_0 w \#)$
- *Kopieren.* $(x, y) = (a, a)$ für alle $a \in \Gamma \cup \{\#\}$
- *Transitionen.*

$$(x, y) = \begin{cases} (qa, pb) & \text{, falls } (q, a, p, b, N) \in \Delta \\ (qa, bp) & \text{, falls } (q, a, p, b, R) \in \Delta \\ (cqa, pcb) & \text{, falls } (q, a, p, b, L) \in \Delta, \text{ für alle } b \in \Gamma \end{cases}$$

- *Leerzeichen einfügen.* $(x, y) = (\#, \square\#)$ und $(x, y) = (\#, \#\square)$
- *Löschen.* $(x, y) = (aq, q)$ und $(x, y) = (qa, q)$ für alle $a \in \Gamma$ und $q \in F$
- *Abschluss.* $(q\#\#, \#)$ für alle $q \in F$.

Es gilt dann $(\mathcal{M}, w) \in \text{HP}$ genau dann, wenn $\mathcal{I}(\mathcal{M}, w) \in \text{MPCP}$. Und gilt $\mathcal{I}(\mathcal{M}, w) = ((x_1, y_1), \ldots, (x_n, y_n)) \in \text{MPCP}$, so besitzt $\mathcal{I}(\mathcal{M}, w)$ eine MPCP-Lösung $(i_1, \ldots, i_m)$ mit $m \geq 0$ und $2 \leq i_1, \ldots, i_m \leq n$. Denn ist $(i_1, \ldots, i_m)$ eine MPCP-Lösung mit $j := \min\{k \mid i_k = 1, 1 \leq k \leq m\}$, so gilt:

$$x_1 x_{i_1} \ldots x_{i_{j-1}} \$ x_{i_{j+1}} \ldots x_{i_m} = y_1 y_{i_1} \ldots y_{i_{j-1}} \$ \square q_0 w \# y_{i_{j+1}} \ldots y_{i_m}.$$

Da das Symbol \$ ausschließlich im ersten Wortpaar vorkommt, muss

$$x_1 x_{i_1} \ldots x_{i_{j-1}} = y_1 y_{i_1} \ldots y_{i_{j-1}}$$

gelten, also ist $(i_1, \ldots, i_{j-1})$ eine MPCP-Lösung, sodass $2 \leq i_1, \ldots, i_{j-1} \leq n$ gilt.

Des Weiteren hängt in $\mathcal{I}(\mathcal{M}, w)$ lediglich das erste Wortpaar von $w \in \Sigma^*$ ab - die restlichen Wortpaare hängen dagegen ausschließlich von $\mathcal{M}$ ab. Sind also $v, w \in \Sigma^*$ verschieden, so unterscheiden sich $\mathcal{I}(\mathcal{M}, v)$ und $\mathcal{I}(\mathcal{M}, w)$ ausschließlich im ersten Wortpaar.

Mithilfe dieser Konstruktion soll nun das folgende Problem, welches im Weiteren als TPCP[4] bezeichnet wird, definiert werden:

Eingabe: Eine deterministische Turingmaschine $\mathcal{M} = (Q, \Sigma, \Gamma, \Delta, q_0, \square, F)$.
Frage: Besitzt $\mathcal{I}(\mathcal{M}, w) = ((x_1, y_1), \ldots, (x_n, y_n))$ für alle $w \in \Sigma^*$ eine Folge $(i_1, \ldots, i_m)$ mit $m \geq 0$ und $2 \leq i_1, \ldots, i_m \leq n$, sodass $(1, i_1, \ldots, i_m)$ eine PCP-Lösung ist? Mit anderen Worten: Gilt für alle $w \in \Sigma^*$: $\mathcal{I}(\mathcal{M}, w) \in \text{MPCP}$?

Da nach [Koz06] das Problem TOTAL, dessen Definition nachfolgend angegeben ist, Π_2^0-vollständig ist, ist es auch leicht zu sehen, dass auch TPCP Π_2^0-hart ist:

Eingabe: Eine deterministische Turingmaschine $\mathcal{M} = (Q, \Sigma, \Gamma, \Delta, q_0, \square, F)$.
Frage: Hält $\mathcal{M}$ bei allen Eingaben $w \in \Sigma^*$ nach endlich vielen Schritten? Mit anderen Worten: Gilt für alle $w \in \Sigma^*$: $(\mathcal{M}, w) \in \text{HP}$?

[4] TPCP ist eine Abkürzung für „**T**otales **PCP**“.

Das komplementäre Problem von TPCP soll nun auf SAT^{aut} mit höchstens endlich vielen (bzw. sogar mit höchstens einer) unendlichen Klauseln reduziert werden.

Proposition 5.11. *Das Problem* SAT^{aut} *mit* höchstens einer *unendlichen Klausel ist* Σ_2^0*-hart.*

Beweis. Sei also $\mathcal{M} = (Q, \Sigma, \Gamma, \Delta, q_0, \square, F)$ eine deterministische Turingmaschine. Definiere $\Pi := \Gamma \uplus Q \uplus \{\#, \$\}$ und $((x_1, y_1), \ldots, (x_n, y_n)) := \mathcal{I}(\mathcal{M}, w)$ für ein beliebiges $w \in \Sigma^*$. Konstruiere analog zum Beweis von Proposition 5.7 nun den folgenden Graphen $\mathcal{K}'(\mathcal{M}) = (A, P, C, N)$ mit den Eigenschaften:

- $A := (\Pi^*)^2$,
- $C := \Pi^* \cup (\{1, \ldots, n\} \times (\Pi^*)^2)$,
- $P := \{((\$, \$\square q_0 w\#), \varepsilon) \mid w \in \Sigma^*\} \cup \left\{((ux_i, vy_i), (i, u, v)) \;\middle|\; \begin{array}{c} u, v \in \Pi^*, \\ 2 \leq i \leq n \end{array}\right\}$,
- $N := \{((u, u), u) \mid u \in \Pi^+\} \cup \{((u, v), (i, u, v)) \mid u, v \in \Pi^*, 2 \leq i \leq n\}$.

Anmerkung. Im Wesentlichen unterscheiden sich $\mathcal{K}'(\mathcal{M})$ und $\mathcal{K}(\mathcal{I}(\mathcal{M}, w))$ nur in der Klausel ε, welche in $\mathcal{K}'(\mathcal{M})$ unendlich ist und alle Anwendungen des ersten Wortpaares enthält und in $\mathcal{K}(\mathcal{I}(\mathcal{M}, w))$ nur das Atom $(\varepsilon, \varepsilon)$ enthält. Die weiteren Klauseln sind bis auf die mehrmalige Anwendung des ersten Wortpaares gleich.

Behauptung 1. Gilt $\mathcal{M} \notin \text{TPCP}$, so existiert eine Menge $S \subseteq A$, sodass für alle $c \in C$ ein $a \in S$ existiert mit $(a, c) \in P$ oder ein $a \in A \setminus S$ existiert mit $(a, c) \in N$.
Beweis. Es gilt also $\mathcal{M} \notin \text{TPCP}$, d.h. es gibt ein $w \in \Sigma^*$ mit $\mathcal{I}(\mathcal{M}, w) \notin \text{MPCP}$. Definiere induktiv über $j \geq 0$ die Mengen $S_j \subseteq A$ wie folgt:

(IA) $j = 0$. $S_0 := \{(\$, \$\square q_0 w\#)\} \subseteq A$.

(IS) $j > 0$. $S_j := \{(ux_i, vy_i) \mid (u, v) \in S_{j-1}, 2 \leq i \leq n\} \subseteq A$.

Dann sei $S := \bigcup_{j \geq 0} S_j \subseteq A$ die zu untersuchende Menge in $\mathcal{K}'(\mathcal{M})$.

Sei nun $c \in C$ eine beliebige Klausel. Dann tritt einer der folgenden Fälle ein:

(Fall 1) $c = \varepsilon$.

Es gilt $(\$, \$\square q_0 w\#) \in S_0 \subseteq S$. Zudem gilt $((\$, \$\square q_0 w\#), \varepsilon) \in P$ nach der Definition von P. Diese Klausel hat also die gewünschte Eigenschaft.

(Fall 2) $c \neq \varepsilon$.

Dieser Fall ist analog zu den Fällen 2 und 3 aus Behauptung 1 im Beweis von Proposition 5.7.

Somit erfüllt $S \subseteq A$ die gewünschten Bedingungen aus SAT^{aut}. q.e.d.

Behauptung 2. Existiert eine Menge $S \subseteq A$, sodass für alle $c \in C$ ein $a \in S$ existiert mit $(a, c) \in P$ oder ein $a \in A \setminus S$ existiert mit $(a, c) \in N$, so gilt auch $\mathcal{M} \notin \text{TPCP}$.
Beweis. Sei nun $S \subseteq A$ eine Menge, die die Bedingungen aus SAT^{aut} erfüllt. Da die Klausel ε nur positive Literale enthält, muss also ein $w \in \Sigma^*$ existieren mit $(\$, \$\square q_0 w\#) \in S$. Es kann nun erneut per Induktion über $m \geq 0$ gezeigt werden, dass für jede endliche Indexfolge $(i_1, \ldots, i_m)$ mit $2 \leq i_1, \ldots, i_m \leq n$ gilt:

$$(\$x_{i_1} \ldots x_{i_m}, \$\square q_0 w\# y_{i_1} \ldots y_{i_m}) \in S.$$

Der Beweis hierfür ist analog zum Induktionsbeweis in Behauptung 2 des Beweises von Proposition 5.7.

Angenommen, die Instanz $\mathcal{I}(\mathcal{M}, w)$ würde nun eine MPCP-Lösung $(i_1, \ldots, i_m)$ mit $m \geq 0$ und $2 \leq i_1, \ldots, i_m \leq n$ besitzen. Es muss also gelten:

$$\$x_{i_1} \ldots x_{i_m} = \$\square q_0 w\# y_{i_1} \ldots y_{i_m}.$$

Die Klausel $\$x_{i_1} \ldots x_{i_m} \in C$ enthält jedoch ausschließlich das Literal $(\$x_{i_1} \ldots x_{i_m}, \$\square q_0 w\# y_{i_1} \ldots y_{i_m}) \in A$ negativ. Da S jedoch die Bedingungen von SAT^{aut} erfüllt, muss auch dieses Literal auch in $A \setminus S$ enthalten sein. Dies steht jedoch im Widerspruch zur Aussage oben, dass dieses Literal in S enthalten ist.

Somit gibt es also keine Indexfolge $(i_1, \ldots, i_m)$ mit $m \geq 0$, die MPCP-Lösung von $\mathcal{I}(\mathcal{M}, w)$ ist. Es gilt also $\mathcal{M} \notin \text{TPCP}$. q.e.d.

Aus den Behauptungen 1 und 2 ergibt sich schließlich: Es gilt genau dann $\mathcal{M} \notin \text{TPCP}$, wenn in $\mathcal{K}'(\mathcal{M})$ eine Menge $S \subseteq A$ mit den Eigenschaften aus SAT^{aut} existiert. Es muss nur noch gezeigt werden, dass $\mathcal{K}'(\mathcal{M})$ aus der Turingmaschine $\mathcal{M}$ berechnet werden kann und automatisch ist.

Behauptung 3. Der Graph $\mathcal{K}'(\mathcal{M})$ ist effektiv automatisch.
Beweis. Die Instanz $\mathcal{I}(\mathcal{M}, w)$ für ein beliebiges $w \in \Sigma^*$ kann aus $\mathcal{M}$ berechnet werden. Die Mengen A, C, N sowie die zweite Teilmenge von P sind nach Behauptung 2 aus dem Beweis von Proposition 5.7 effektiv automatisch. Es muss also nur noch die erste Teilmenge von P betrachtet werden. Diese wird von einem Dreiband-Automaten akzeptiert, der wie folgt funktioniert: Auf dem ersten bzw. dritten Band wird nur das Symbol \$ bzw. das leere Wort gelesen. Auf dem zweiten Band dagegen wird zunächst $\$\square q_0$ gelesen, anschließend ein beliebiges Wort aus Σ^* und zum Schluss noch einmal das Symbol #. Dieser Automat kann aus $\mathcal{M}$ berechnet werden. Damit ist auch die Menge P nach den Abschlusseigenschaften automatischer Relationen effektiv automatisch. q.e.d.

Es folgt also: $\mathcal{M} \notin \text{TPCP}$ genau dann, wenn $\mathcal{K}'(\mathcal{M}) \in \text{SAT}^{\text{aut}}$. Da das Komplement von TPCP nach der Bemerkung oben Σ_2^0-hart ist, ist auch SAT^{aut} Σ_2^0-hart. ■

Aus den Propositionen 5.10 und 5.11 folgt schließlich unmittelbar:

Satz 5.12. *Das Problem* SAT^{aut} *mit der Einschränkung, dass nur endlich vielen Klauseln unendlich viele Literale zugeordnet werden, ist* Σ_2^0*-vollständig.*

Auch an dieser Stelle stellt sich noch die Frage, in welcher Klasse der arithmetischen Hierarchie sich SAT^{∞} mit endlich vielen unendlichen Klauseln einordnet.

Sei also $\mathcal{G} = (A, C, P, N)$ wie in SAT^{∞} mit der gewünschten Einschränkung. Dann ist $\mathcal{G} \in \text{SAT}^{\infty}$ genau dann, wenn gilt: Es existieren *endliche*, disjunkte Mengen $S_+, S_- \subseteq A$, sodass gilt:

$$\forall c \in C\colon \left(c \text{ ist endliche Klausel } \vee \bigvee_{a \in S_+} P(a,c) \vee \bigvee_{a \in S_-} N(a,c) \right)$$

und der Graph $\mathcal{G}'$, der durch Entfernen der Literale aus S_+ und S_- sowie der Klauseln, in denen ein $a \in S_+$ positiv bzw. ein $a \in S_-$ negativ vorkommt, entsteht, ist in SAT^{∞} mit ausschließlich endlichen Klauseln.

Die Frage, ob $c \in C$ endlich ist, ist Σ_2^0-vollständig [Koz06]. Die letztgenannte Frage ist nach der Begründung in Abschnitt 5.2 in Π_2^0. Damit ergibt sich schließlich, dass SAT^{∞} mit endlich vielen unendlichen Klauseln in Σ_4^0 ist. Aber auch hier ist nicht bekannt, ob dieses Problem auch Σ_4^0-hart ist.

Kapitel 6

Graphfärbungen

In diesem Kapitel soll abschließend nun das in der Einleitung erwähnte Färbungsproblem von Graphen untersucht werden. Zunächst soll noch einmal der Begriff einer Färbung definiert werden:

Definition 6.1. Sei $\mathcal{G} = (V, E)$ ein Graph und C eine Menge von Farben. Eine ***C*-Färbung** ist eine Funktion $c\colon V \to C$, sodass für alle Knoten $u, v \in V$ mit $(u, v) \in E$ gilt: $c(u) \neq c(v)$. Ein Graph $\mathcal{G}$ heißt ***C*-färbbar**, wenn $\mathcal{G}$ eine C-Färbung besitzt.

Sei $k \in \mathbb{N} \setminus \{0\}$. Ein Graph $\mathcal{G}$ heißt ***k*-färbbar**, wenn $\mathcal{G}$ C-färbbar für eine Menge C mit $|C| = k$ ist.

Es ergibt sich schließlich das folgende natürliche Graphproblem:

Eingabe: Ein rekursiver Graph $\mathcal{G} = (V, E)$, eine rekursive Menge von Farben C und eine Farbe $c_0 \in C$.
Frage: Ist $\mathcal{G}$ C-färbbar, wobei die Farbe c_0 unendlich oft verwendet wird?

Dieses Problem wird im Folgenden auch $\textsc{Color}^{\infty}$ genannt. Die automatische Variante dieses Problems - im Weiteren $\textsc{Color}^{\text{aut}}$ genannt - ist dann das folgende Problem:

Eingabe: Ein *automatischer* Graph $\mathcal{G} = (V, E)$, eine *reguläre* Menge von Farben C und eine Farbe $c_0 \in C$.
Frage: Ist $\mathcal{G}$ C-färbbar, wobei die Farbe c_0 unendlich oft verwendet wird?

6.1 Der unendliche Fall

Zunächst soll der Fall betrachtet werden, dass die Farbmenge C unendlich ist. Hirst und Harel haben in [HH96, Proposition 8] gezeigt, dass $\textsc{Color}^{\infty}$ Σ^1_1-vollständig ist. Für automatische Graphen soll dagegen die Entscheidbarkeit gezeigt werden:

Proposition 6.2. *Das Problem* $\textsc{Color}^{\text{aut}}$ *mit unendlicher Farbmenge C ist entscheidbar. Falls eine Färbung mit unendlicher Farbmenge existiert, so kann eine solche berechnet werden.*

Beweis. Sei $\mathcal{G} = (V, E)$ ein automatischer Graph und C reguläre, unendliche Farbmenge. COLOR$^{\text{aut}}$ mit unendlicher Farbmenge soll nun auf IND SET$^{\text{aut}}$ reduziert werden.

Behauptung 1. Existiert eine unendliche Menge $U \subseteq V$ unabhängiger Knoten, so gilt $(\mathcal{G}, C, c_0) \in$ COLOR$^{\text{aut}}$.
Beweis. Sei $U \subseteq V$ eine unendliche Menge unabhängiger Knoten. Betrachte die folgende Färbung:

$$c\colon V \to C\colon v \mapsto \begin{cases} c_0 & \text{, falls } v \in U \\ c'(v) & \text{, falls } v \notin U, \end{cases}$$

wobei $c'\colon V \setminus U \to C \setminus \{c_0\}$ eine beliebige injektive Abbildung ist. Seien nun $u, v \in V$ zwei beliebige Knoten mit $(u, v) \in E$. Es gilt nicht gleichzeitig $u \in U$ und $v \in U$, d.h. auch nicht gleichzeitig $c(u) = c_0$ und $c(v) = c_0$. O.B.d.A. gelte nun also $c(u) \neq c_0$. Da die Abbildung c' injektiv auf $C \setminus \{c_0\}$ abbildet, ist dann $c(u) = c'(u) \neq c(v)$. Also ist c eine korrekte Färbung. q.e.d.

Behauptung 2. Gilt $(\mathcal{G}, C, c_0) \in$ COLOR$^{\text{aut}}$, dann existiert eine unendliche Menge $U \subseteq V$ unabhängiger Knoten.
Beweis. Sei nun $c\colon V \to C$ eine Färbung, sodass $|c^{-1}(c_0)| = \infty$. Dann gilt für alle Knoten $u, v \in c^{-1}(c_0)$: $(u, v) \notin E$ (andernfalls wäre dies keine legale Färbung). Damit ist $c^{-1}(c_0)$ eine unendliche Menge von unabhängigen Knoten. q.e.d.

Mit den Behauptungen 1 und 2 wurde schließlich die Reduktion von COLOR$^{\text{aut}}$ mit unendlicher Farbmenge C auf IND SET$^{\text{aut}}$ gezeigt. IND SET$^{\text{aut}}$ ist nach Korollar 3.4 entscheidbar. Damit ist die erste Aussage der Proposition bewiesen.

Behauptung 3. Gilt $(\mathcal{G}, C, c_0) \in$ COLOR$^{\text{aut}}$, so kann eine solche Färbung $c\colon V \to C$ berechnet werden.
Beweis. Sei $\mathcal{G}$ C-färbbar. Zudem sei $U \subseteq V$ eine unendliche Menge unabhängiger Knoten (diese existiert nach Behauptung 2 und eine solche reguläre Menge kann nach Korollar 3.4 berechnet werden). Dann berechnet der folgende Algorithmus eine Graphfärbung:

Eingabe: Knoten $v \in V$
(1) **if** $v \in U$ **then**
(2) | **return** c_0
(3) **else**
(4) | Sei $v_1, v_2, \ldots$ die längenlexikografische Aufzählung von $V \setminus U$ und
| $c_1, c_2, \ldots$ die längenlexikografische Aufzählung von $C \setminus \{c_0\}$.
(5) | Suche $i \geq 1$ mit $v_i = v$.
(6) | **return** c_i
(7) **end if**

Da die längenlexikografische Aufzählung von V und C automatisch ist, kann also für einen Knoten $v \in V$ ein $i \geq 1$ mit $v = v_i$ bestimmt werden (teste zuerst v_1, dann v_2 usw.). Damit ist dies tatsächlich ein Algorithmus. Die berechnete Funktion

ist tatsächlich eine C-Färbung, da alle Knoten $v \in U$ die Farbe c_0 erhalten (U enthält nur unabhängige Knoten) und für jede Farbe $c \in C \setminus \{c_0\}$ genau ein Knoten $v \in V \setminus U$ existiert, sodass der Algorithmus c ausgibt. Damit ist dies eine Abbildung $c\colon V \to C$ wie oben beschrieben. q.e.d.

Damit ist auch die zweite Aussage aus der Proposition bewiesen. ∎

Die berechnete Färbung ist also rekursiv, aber nicht automatisch. Es stellt sich hierbei natürlich die Frage, ob auch ein endlicher Automat existiert, der eine solche Färbung akzeptiert. Das folgende Beispiel soll zeigen, dass diese Frage im Allgemeinen verneint werden kann. Hierfür wird die folgende aus der Analysis bekannte Ungleichung verwendet:

$$2^x > x + 1 \text{ für } x > 1. \tag{6.1}$$

Sei $V := \{0,1\}^* \cup \{2\}^+$ und $E := V^2 \setminus (\{2\}^+)^2$. Dann ist $\mathcal{G} = (V, E)$ ein automatischer Graph, wobei der durch die Knotenmenge $\{2\}^+$ induzierte Teilgraph diskret ist. Sei weiterhin $C := \{1\}^*$ und $c_0 \in C$ beliebig. Der Graph $\mathcal{G}$ ist dann offensichtlich C-färbbar:

$$c\colon V \to C\colon v \in C \mapsto \begin{cases} c_0 & \text{, falls } v \in \{2\}^+ \\ c'(v) & \text{, falls } v \in \{0,1\}^*, \end{cases}$$

wobei $c'\colon \{0,1\}^* \to C \setminus \{c_0\}$ eine beliebige injektive Funktion ist.

Sei nun $c\colon V \to C$ eine beliebige C-Färbung mit $|c^{-1}(c_0)| = \infty$. Es soll nun gezeigt werden, dass es keinen endlichen Automaten $\mathcal{A}$ gibt mit $L(\mathcal{A}) = \otimes c$.

Angenommen, es gäbe einen solchen Automaten $\mathcal{A}$. Zunächst muss gelten: $c^{-1}(c_0) \subseteq \{2\}^+$. Denn würde ein $v \in \{0,1\}^*$ mit $c(v) = c_0$ existieren, so gilt für jedes $w \in V \setminus \{v\}$ $(v, w) \in E$ und damit $c(w) \neq c(v) = c_0$. Also würde $c^{-1}(c_0) = \{v\}$ gelten - im Widerspruch zur Definition von c. Damit ist auch $c' := c \cap (\{0,1\}^* \times C)$ injektiv und nach den Abschlusseigenschaften regulärer Sprachen automatisch. Sei $\mathcal{A}'$ ein Automat mit $L(\mathcal{A}') = c'$ und sei $m := \max\{2, m'\}$, wobei m' die Konstante aus Lemma 2.7 ist. Betrachte die Menge aller Wörter über dem Alphabet $\{0,1\}$ mit der Länge 2^m: Es gilt $|\{0,1\}^{2^m}| = 2^{2^m}$. Da c' injektiv ist, muss auch

$$\left|c'\left[\{0,1\}^{2^m}\right]\right| = 2^{2^m}$$

gelten. Da es über dem Alphabet $\{1\}$ jeweils nur genau ein Wort der Länge n mit $n \geq 0$ gibt, sind in $c'[\{0,1\}^{2^m}]$ keine zwei Wörter mit gleicher Wortlänge enthalten. Es gibt also ein Wort $v \in \{0,1\}^{2^m}$ mit Wortlänge $|c'(v)| \geq 2^{2^m} - 1$. Es gilt dann also:

$$\begin{aligned} |c'(v)| - |v| &\geq (2^{2^m} - 1) - 2^m = 2^m \cdot (2^{2^m - m} - 1) - 1 \\ &\overset{(6.1)}{>} 2^m \cdot (2^1 - 1) - 1 \geq 2^m - 1 \overset{(6.1)}{>} m + 1 - 1 = m, \end{aligned}$$

d.h. $|c'(v)| > m + |v|$. Dies steht jedoch im Widerspruch zu Lemma 2.7. Also gibt es keinen Automaten $\mathcal{A}$ mit $L(\mathcal{A}) = \otimes c$. Da jedoch c eine beliebige C-Färbung war, wurde damit gezeigt, dass für den Graphen $\mathcal{G} = (V, E)$ kein Automat existiert, der eine beliebige C-Färbung $c\colon V \to C$ akzeptiert.

6.2 Der endliche Fall

Nachdem nun die Entscheidbarkeit für unendliche Färbungen gezeigt wurde, soll nun noch der Fall mit endlicher Farbmenge betrachtet werden. In [Bea76] wurde gezeigt, dass COLOR^{∞} für endliche Farbmengen C Π_1^0-vollständig ist. Für automatische Graphen und $|C| > 1$ soll ein ähnliches Ergebnis gezeigt werden. Zuvor wird aber noch der Fall $|C| = 1$ betrachtet:

Proposition 6.3. *Das Problem* $\text{COLOR}^{\text{aut}}$ *mit* $|C| = 1$ *ist entscheidbar.*

Beweis. Sei $\mathcal{G} = (V, E)$ ein automatischer Graph. Dann ist $\mathcal{G}$ genau dann 1-färbbar, wenn V unendlich ist und für alle Knoten $u, v \in V$ $(u, v) \notin E$ gilt, d.h. wenn $E = \emptyset$. Das Endlichkeits- und das Leerheitsproblem ist für reguläre Sprachen entscheidbar. ∎

Nun soll der Fall $|C| \geq 2$ betrachtet werden. Hierfür die Π_1^0-Vollständigkeit gezeigt werden. Dass dieses Problem in Π_1^0 ist, folgt aus der Π_1^0-Vollständigkeit im rekursiven Fall.

Es muss also nur noch die Π_1^0-Härte von $\text{COLOR}^{\text{aut}}$ mit zwei Farben gezeigt werden. Daraus folgt dann auch unmittelbar die Π_1^0-Härte für $\text{COLOR}^{\text{aut}}$ mit $|C| \geq 3$: Sei $\mathcal{G} = (V, E)$ ein automatischer Graph. Dann ist $\mathcal{G}' = (V \uplus \{\#\}, E \cup \{(\#, v), (v, \#) \mid v \in V\})$ ebenfalls automatisch. Zudem ist $\mathcal{G}$ genau dann k-färbbar (für $k \geq 1$), wenn $\mathcal{G}'$ $(k+1)$-färbbar ist (dem Knoten $\#$ wird die eine zusätzliche Farbe zugeordnet).

Um die Π_1^0-Härte von $\text{COLOR}^{\text{aut}}$ mit zwei Farben zu zeigen, werden zunächst die folgenden Aussagen benötigt, die hier jedoch nicht bewiesen werden:

Lemma 6.4.

(i) Sei $k \in \mathbb{N} \setminus \{0\}$ *und* $\mathcal{G} = (V, E)$ *ein Graph. Dann gilt:* $\mathcal{G}$ *ist genau dann* k*-färbbar, wenn jeder endliche Teilgraph von* $\mathcal{G}$ k*-färbbar ist. [BE51, Theorem 1]*

(ii) Ein Graph $\mathcal{G}$ *ist genau dann* 2*-färbbar, wenn er bipartit ist. [ADH98]*

(iii) Ein Graph $\mathcal{G}$ *ist genau dann bipartit, wenn er keine Kreise ungerader Länge enthält. [ADH98, Theorem 2.1.3]*

Es soll nun erneut mithilfe einer Reduktion des Komplements von PCP die Π_1^0-Härte von $\text{COLOR}^{\text{aut}}$ mit zwei Farben gezeigt werden.

Proposition 6.5. *Das Problem* $\textsc{Color}^{\text{aut}}$ *mit* $|C| = 2$ *ist* Π_1^0*-vollständig.*

Beweis. Verwende wie in Abschnitt 5.2 das Komplement von PCP, um die Π_1^0-Härte zu zeigen.

Idee. Konstruiere aus $I = ((x_1, y_1), \ldots, (x_n, y_n))$ einen Graphen bestehend aus (disjunkten!) Teilgraphen der folgenden Form:

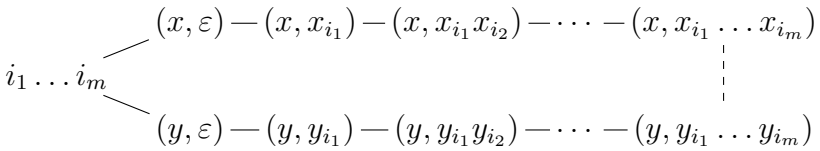

Abbildung 6.1.

Hierbei sind x und y zwei unterschiedliche (neue) Symbole. Des Weiteren ist $(i_1, \ldots, i_m)$ eine beliebige Indexfolge mit $m > 0$ und $1 \leq i_1, \ldots, i_m \leq n$. Die gestrichelte Kante zwischen den Knoten $(x, x_{i_1} \ldots x_{i_m})$ und $(y, y_{i_1} \ldots y_{i_m})$ wird genau dann eingefügt, wenn $x_{i_1} \ldots x_{i_m} = y_{i_1} \ldots y_{i_m}$ gilt.

Der konstruierte Graph besitzt dann einen Kreis ungerader Länge und ist damit nicht 2-färbbar genau dann, wenn I eine PCP-Lösung besitzt.

Sei also $I = ((x_1, y_1), \ldots, (x_n, y_n))$ mit $n \geq 1$ eine endliche Folge von Wortpaaren über dem Alphabet Σ. Konstruiere nun den Graphen $\mathcal{C}(I) = (V, E)$ über dem Alphabet $\Gamma := \Sigma \cup \{1, \ldots, n\} \cup \{x, y\}$ mit den folgenden Eigenschaften:

- V ist die Vereinigung der folgenden Mengen:
  - $\{x\} \times \{(u, v) \in (\{1, \ldots, n\}^*)^2 \mid v \text{ ist Präfix von } u\} \times \{x_1, \ldots, x_n\}^*$,
  - $\{y\} \times \{(u, v) \in (\{1, \ldots, n\}^*)^2 \mid v \text{ ist Präfix von } u\} \times \{y_1, \ldots, y_n\}^*$ und
  - $\{1, \ldots, n\}^*$.
- $E := E' \cup E'^{-1}$, wobei E' die Vereinigung der folgenden Mengen ist:
  - $E_x := \left\{ ((x,u,v,w),(x,u,vi,wx_i)) \,\middle|\, \begin{array}{c} vi \text{ Präfix von } u \in \{1,\ldots,n\}^*, \\ i \in \{1,\ldots,n\}, w \in \{x_1,\ldots,x_n\}^* \end{array} \right\}$ (die Kanten des oberen Pfades in Abbildung 6.1),
  - $E_y := \left\{ ((y,u,v,w),(y,u,vi,wy_i)) \,\middle|\, \begin{array}{c} vi \text{ Präfix von } u \in \{1,\ldots,n\}^*, \\ i \in \{1,\ldots,n\}, w \in \{y_1,\ldots,y_n\}^* \end{array} \right\}$ (die Kanten des unteren Pfades in Abbildung 6.1),
  - $E_i := \{(u, (x, u, \varepsilon, \varepsilon)), (u, (y, u, \varepsilon, \varepsilon)) \mid u \in \{1, \ldots, n\}^*\}$ (die Kanten zum Knoten $i_1 \ldots i_m$ in Abbildung 6.1) und

– $E_f := \left\{ ((x, v, v, w), (y, v, v, w)) \;\middle|\; \begin{array}{c} v \in \{1, \ldots, n\}^*, \\ w \in \{x_1, \ldots, x_n\}^+ \cap \{y_1, \ldots, y_n\}^+ \end{array} \right\}$
(die gestrichelte Kante in Abbildung 6.1).

Beispiel. Betrachte die PCP-Instanz $I := ((0, 010), (1, 101), (0101, 01))$. Nachfolgend ist ein Ausschnitt des konstruierten Graphen $\mathcal{C}(I)$ abgebildet:

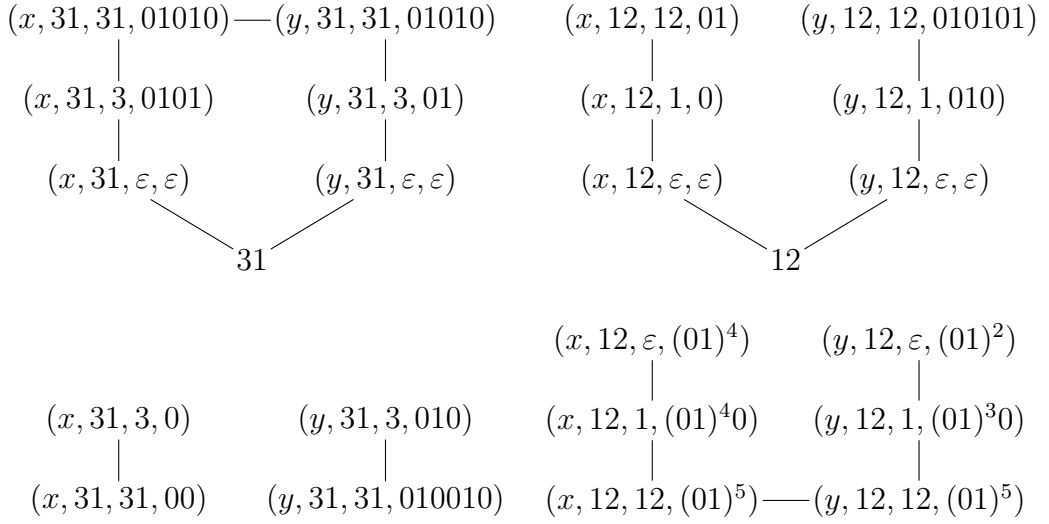

Abbildung 6.2. Ausschnitt des Graphen $\mathcal{C}(I)$.

Die Instanz I besitzt die Lösung $(3, 1)$. In der Abbildung 6.2 links oben befindet sich ein Kreis ungerader Länge, also ist $\mathcal{C}(I)$ nicht 2-färbbar. Rechts oben wird beispielhaft ein Teilgraph mit einer Indexfolge dargestellt, die keine PCP-Lösung ist. Darunter werden beispielhaft weitere Knoten (a, u, v, w) mit $a \in \{x, y\}$ aus dem Graphen dargestellt, in denen $v = i_1 \ldots i_m$ und $w \neq a_{i_1} \ldots a_{i_m}$ gilt.

Behauptung 1. Gilt $I \in \text{PCP}$, so ist $\mathcal{C}(I)$ nicht 2-färbbar.
Beweis. Sei also $I \in \text{PCP}$. Dann existiert eine Indexfolge $(i_1, \ldots, i_m)$ mit $m \geq 1$, $1 \leq i_1, \ldots, i_m \leq n$ und $x_{i_1} \ldots x_{i_m} = y_{i_1} \ldots y_{i_m}$. Dann gilt:

- $(i_1 \ldots i_m, (x, i_1 \ldots i_m, \varepsilon, \varepsilon)), ((y, i_1 \ldots i_m, \varepsilon, \varepsilon), i_1 \ldots i_m) \in E$ nach Definition von E_i.
- $((x, i_1 \ldots i_m, i_1 \ldots i_k, x_1 \ldots x_k), (x, i_1 \ldots i_m, i_1 \ldots i_k i_{k+1}, x_1 \ldots x_k x_{k+1})) \in E$ für alle $0 \leq k < m$ nach Definition von E_x.
- $((y, i_1 \ldots i_m, i_1 \ldots i_k i_{k+1}, y_1 \ldots y_k y_{k+1}), (y, i_1 \ldots i_m, i_1 \ldots i_k, y_1 \ldots y_k)) \in E$ für alle $0 \leq k < m$ nach Definition von E_y.
- $((x, i_1 \ldots i_m, i_1 \ldots i_m, x_1 \ldots x_m), (y, i_1 \ldots i_m, i_1 \ldots i_m, y_1 \ldots y_m)) \in E$, da nach Voraussetzung $x_1 \ldots x_m = y_1 \ldots y_m$ gilt und nach Definition von E_f.

Damit enthält $\mathcal{C}(I)$ also den folgenden Kreis:

$$(i_1 \dots i_m, (x, i_1 \dots i_m, \varepsilon, \varepsilon), (x, i_1 \dots i_m, i_1, x_1), \dots, (x, i_1 \dots i_m, i_1 \dots i_m, x_{i_1} \dots x_{i_m}),$$
$$(y, i_1 \dots i_m, i_1 \dots i_m, y_{i_1} \dots y_{i_m}), \dots, (y, i_1 \dots i_m, i_1, y_1), (y, i_1 \dots i_m, \varepsilon, \varepsilon), i_1 \dots i_m).$$

Dieser Kreis besteht aus genau $2 \cdot (m+1) + 1$ Knoten und besitzt damit ungerade Länge. Nach Lemma 6.4(iii) ist dann $\mathcal{C}(I)$ nicht bipartit und damit nach Lemma 6.4(ii) nicht zweifärbbar. q.e.d.

Um die Umkehrung von Behauptung 1 zeigen zu können, sind noch die folgenden Beobachtungen notwendig.

Beobachtung 2. Seien $u, v \in \{1, \dots, n\}^+$, v ein echter Präfix von u und $(i_1, \dots, i_k)$ eine beliebige Indexfolge mit $1 \leq i_1, \dots, i_k \leq n$ und $k \geq 0$. Dann gilt:

(i) Der Knoten $(x, u, v, x_{i_1} \dots x_{i_k}) \in V$ hat maximal die folgenden zwei Nachbarknoten in E:

  (a) $(x, u, v', x_{i_1} \dots x_{i_{k-1}})$, falls $v = v' i_k$ gilt.

  (b) $(x, u, vi, x_{i_1} \dots x_{i_k} x_i)$, falls vi Präfix von u und $i \in \{1, \dots, n\}$ ist.

  Analog gilt diese Aussage auch für den Knoten $(y, u, v, y_{i_1} \dots y_{i_k}) \in V$.

(ii) Der Knoten $(x, u, u, x_{i_1} \dots x_{i_k}) \in V$ hat maximal die folgenden zwei Nachbarknoten in E:

  (a) $(y, u, u, y_{j_1} \dots y_{j_l})$, falls $l \geq 1$ und $1 \leq j_1, \dots, j_l \leq n$ existieren, sodass $x_{i_1} \dots x_{i_k} = y_{j_1} \dots y_{y_l}$ gilt.

  (b) $(x, u, u', x_{i_1} \dots x_{i_{k-1}})$, falls $u = u' i_k$ gilt.

  Analog gilt diese Aussage auch für den Knoten $(y, u, u, y_{i_1} \dots y_{i_k}) \in V$.

(iii) Der Knoten $(x, u, \varepsilon, x_{i_1} \dots x_{i_k}) \in V$ mit $k \geq 1$ hat mit $(x, u, i, x_{i_1} \dots x_{i_k} x_i)$, sodass i Präfix von u ist und $i \in \{1, \dots, n\}$, genau einen Nachbarknoten in E. Der Knoten $(x, u, \varepsilon, \varepsilon)$ hat zusätzlich noch u als Nachbarknoten. Analog gilt dies für $(y, u, \varepsilon, y_{i_1} \dots y_{i_k}) \in V$.

Beobachtung 3. Definiere die Knotenmenge V_u für ein $u \in \{1, \dots, n\}^*$ wie folgt:

$$V_u := \left\{ \begin{array}{c} u, (x, u, v, w_x), \\ (y, u, v, w_y) \end{array} \;\middle|\; \begin{array}{c} v \text{ ist Präfix von } u, \\ w_x \in \{x_1, \dots, x_m\}^*, w_y \in \{y_1, \dots, y_m\}^* \end{array} \right\}.$$

Definiere $\mathcal{C}_u(I)$ als den durch V_u induzierten Teilgraphen von $\mathcal{C}(I)$. Dann sind die Knotenmengen V_u und V_w mit $w \in \{1, \dots, n\}^* \setminus \{u\}$ disjunkt und es gibt in E keine Kanten zwischen den Teilgraphen $\mathcal{C}_u(I)$ und $\mathcal{C}_w(I)$.

Behauptung 4. Ist $\mathcal{C}(I)$ nicht 2-färbbar, so gilt $I \in \mathrm{PCP}$.

Beweis. Sei nun also $\mathcal{C}(I)$ nicht zweifärbbar. Nach Lemma 6.4(ii) und (iii) enthält $\mathcal{C}(I)$ dann also einen Kreis ungerader Länge. Sei $(v_0, \ldots, v_k)$ ein solcher Kreis mit ungeradem $k > 1$. Nach Beobachtung 3 gibt es ein $u \in \{1, \ldots, n\}^*$, sodass $v_0, \ldots, v_k \in V_u$. Sei $(i_1, \ldots, i_m)$ die Indexfolge mit $u = i_1 \ldots i_m$. Betrachte nun die folgende Fallunterscheidung:

(Fall 1) $v_0 = u$. Nach Definition von E gilt: $\{v_1, v_{k-1}\} = \{(x, u, \varepsilon, \varepsilon), (y, u, \varepsilon, \varepsilon)\}$, o.B.d.A. gelte $v_1 = (x, u, \varepsilon, \varepsilon)$. Es gilt weiterhin $v_2 = (x, u, i_1, x_{i_1})$ und $v_{k-2} = (y, u, i_1, y_{i_2})$. Nach Beobachtung 2(i) kann nun induktiv gezeigt werden, dass

$$v_{l+1} = (x, u, i_1 \ldots i_l, x_{i_1} \ldots x_{i_l}) \text{ und } v_{k-(l+1)} = (y, u, i_1 \ldots i_l, y_{i_1} \ldots y_{i_l})$$

für alle $2 \leq l \leq m$ gelten muss. Insbesondere muss gelten:

$$v_{m+1} = (x, u, u, x_{i_1} \ldots x_{i_m}) \text{ und } v_{k-(m+1)} = (y, u, u, y_{i_1} \ldots y_{i_m}).$$

Da $(v_0, \ldots, v_k)$ ein Kreis ist, muss nach Beobachtung 2(ii) dann

$$v_{m+2} = (y, u, u, y_{i_1} \ldots y_{i_m}) = v_{k-(m+1)}$$

gelten (und damit $k = 2 \cdot m + 3$). Nach Definition von E muss hierzu $x_{i_1} \ldots x_{i_m} = y_{i_1} \ldots y_{i_m}$ gelten. Dies ist die Definition einer PCP-Lösung für I, also gilt $I \in \text{PCP}$.

(Fall 2) $v_0 = (x, u, u, w)$. Da $(v_0, \ldots, v_k)$ ein Kreis ist, müssen alle diese Knoten mindestens zwei Nachbarknoten besitzen. Somit gilt nach Beobachtung 2(ii) $\{v_1, v_{k-1}\} = \{(y, u, u, w), (x, u, u_m, w_m)\}$ mit $u = u_m i_m$ und $w = w_m x_{i_m}$, o.B.d.A. gelte $v_1 = (x, u, u_m, w_m)$. Mithilfe von Beobachtung 2(i) kann dann induktiv gezeigt werden, dass $v_{l+1} = (x, u, u_{m-l}, w_{m-l})$ mit $u = u_{m-l} i_{m-l} \ldots i_m$ und $w = w_{m-l} x_{i_{m-l}} \ldots x_{i_m}$ für alle $0 \leq l \leq m$. Insbesondere gilt dann $v_{m+1} = (x, u, \varepsilon, w_0)$. Da jedoch v_{m+1} ebenfalls mindestens zwei Nachbarknoten besitzen muss, gilt nach Beobachtung 2(iii) $w_0 = \varepsilon$. Nach Definiton von E muss dann $v_{m+2} = u$ gelten. Durch eine Indexverschiebung entsteht der folgende Kreis:

$$(v'_0 = v_{m+2}, v'_1 = v_{m+3}, \ldots, v'_{k-(m+3)} = v_{k-1}, v'_{k-(m+2)} = v_0, \ldots, v'_k = v_{m+2})$$

und nach Fall 1 gilt dann $I \in \text{PCP}$.

(Fall 3) $v_0 = (x, u, v, w)$ mit v ist echter Präfix von w. Nach ähnlicher Argumentation wie im Fall 2 kann eine Indexverschiebung vorgenommen werden, sodass $v'_0 = u$ gilt und damit $I \in \text{PCP}$.

(Fall 4) $v_0 = (y, u, v, w)$. Die Argumentation erfolgt analog zu den Fällen 2 und 3. Es gilt schließlich ebenfalls $I \in \text{PCP}$. q.e.d.

Aus den Behauptungen 1 und 4 folgt somit, dass I genau dann keine PCP-Lösung besitzt, wenn $\mathcal{C}(I)$ 2-färbbar ist. Es muss nur noch gezeigt werden, dass $\mathcal{C}(I)$ auch automatisch ist und aus I berechnet werden kann.

Behauptung 5. Der Graph $\mathcal{C}(I)$ ist effektiv automatisch.
Beweis. Die erste Teilmenge von V kann von einem Vierband-Automaten akzeptiert werden, der wie folgt rechnet: Auf dem ersten Band wird ausschließlich das Symbol x gelesen, auf dem Band 2 ein beliebiges Wort über $\{1, \ldots, n\}^*$. Band 3 liest zunächst den gleichen Inhalt wie auf Band 2 und liest anschließend nur noch $\diamond$-Symbole. Schlussendlich wird auf Band 4 ein Wort aus $\{x_1, \ldots, x_n\}^*$ gelesen. Letztere Sprache ist nach den Abschlusseigenschaften regulär und kann aus I berechnet werden. Somit sind die ersten beiden Teilmengen von V automatisch (die zweite Teilmenge wird von einem analogen Automaten akzeptiert). Offensichtlich ist auch die dritte Teilmenge regulär. Nach den Abschlusseigenschaften ist also V effektiv automatisch.

Die Menge E_x lässt sich nochmals in n Teilmengen unterteilen (diese unterscheiden sich im $i \in \{1, \ldots, n\}$, welches in der dritten Komponente hinzugefügt wird). Eine solche Teilmenge mit $i \in \{1, \ldots, n\}$ wird von dem folgenden Achtband-Automaten akzeptiert: Lese auf Band 1 und 5 ausschließlich das Symbol x. Auf den Bändern 2 und 6 wird synchron das gleiche Wort aus $\{1, \ldots, n\}^*$ gelesen, auf Band 7 ein Präfix dieses Wortes, welcher mit i endet, und auf Band 3 derselbe Präfix ohne das endende i. Auf Band 4 wird noch ein beliebiges Wort aus $\{x_1, \ldots, x_n\}^*$ gelesen und auf Band 8 zunächst dasselbe Wort und anschließend noch das Wort x_i. Somit ist E_x und analog auch E_y effektiv automatisch. Die Menge E_i wird von einem Fünfband-Automaten akzeptiert, der auf dem ersten und dritten Band ein und dasselbe Wort aus $\{1, \ldots, n\}^*$, auf Band zwei ausschließlich x bzw. y und auf den restlichen Bändern nichts liest. Die Menge E_f wird ebenfalls von einem Achtband-Automaten akzeptiert, der auf Band 1 nur x und auf Band 5 nur y, auf den Bändern 2, 3, 6 und 7 ein und dasselbe Wort aus $\{1, \ldots, n\}^*$ liest und auf den restlichen Bändern ein und dasselbe Wort aus der effektiv regulären Menge $\{x_1, \ldots, x_n\}^+ \cap \{y_1, \ldots, y_n\}^+$. Nach den Abschlusseigenschaften ist damit E' automatisch. Nach dem Fundamentalsatz ist dann auch E effektiv automatisch. q.e.d.

Damit wurde gezeigt, dass I genau dann keine PCP-Lösung besitzt, wenn $(\mathcal{C}(I), \{c_0, c_1\}, c_0) \in \textsc{Color}^{\text{aut}}$. Da das erstgenannte Problem nach [Pos46] Π_1^0-hart ist, ist dies auch $\textsc{Color}^{\text{aut}}$ mit $|C| = 2$. Und da dieses Problem in Π_1^0 enthalten ist, folgt dann auch die Π_1^0-Vollständigkeit. ∎

Aus den Propositionen 6.2, 6.3 und 6.5 lässt sich schließlich folgern:

Satz 6.6. *Das Problem* $\textsc{Color}^{\text{aut}}$ *ist* Π_1^0*-vollständig.*

Kapitel 7

Zusammenfassung

Es wurden die Entscheidbarkeiten der folgenden Probleme für rekursive und automatische Graphen untersucht:

(1) Existenz einer unendlichen unabhängigen Knotenmenge, Existenz eines unendlichen Pfades in Ordnungsbäumen, Existenz einer Mengenüberdeckung und das Mengenpackungsproblem,

(2) Existenz einer exakten Überdeckung bzw. eines perfekten Matchings und Erfüllbarkeit unendlicher aussagenlogischer Formeln in Konjunktiver Normalform,

(3) Existenz einer Graphfärbung.

Nach Hirst und Harel [HH96] sind all diese Probleme für rekursive Graphen Σ^1_1-vollständig. Für die automatischen Versionen dieser Probleme ließen sich die folgenden Ergebnisse zeigen:

Für die in (1) genannten Probleme ergab sich die Entscheidbarkeit mithilfe der Entscheidbarkeit der FSO-Logik. Weitherhin konnte gezeigt werden, dass sich für alle diese Probleme auch reguläre Mengen berechnen lassen, die die gewünschten Anforderungen erfüllen.

Im Gegensatz dazu wurde gezeigt, dass die Probleme aus (2) nicht entscheidbar sind und sich auf der gleichen Stufe in der analytischen Hierarchie befinden, wie die entsprechenden rekursiven Versionen. Durch diverse Einschränkungen dieser Probleme kann jedoch auch gezeigt werden, dass sich diese auf niedrigen Stufen in der arithmetischen Hierarchie einordnen lassen.

Letztendlich wurde für das in (3) genannte Problem ebenfalls die Unentscheidbarkeit gezeigt. Dennoch ließ sich auch dieses Problem mit Π^0_1 in einer niedrigen Stufe der arithmetischen Hierarchie einordnen. Für den im rekursiven Fall besonders komplizierten Fall mit unendlicher Farbmenge konnte sogar die Entscheidbarkeit in den automatischen Graphen gezeigt werden.

Die folgende Tabelle fasst alle Ergebnisse dieser Arbeit in einer Übersicht zusammen. Hierbei werden rekursive und automatische Graphen miteinander verglichen und die hier angewendeten Reduktionen angegeben.

Problem	**rekursive Graphen**	**automat. Graphen**	**Reduktion**
Unabhängige Knotenmenge	Σ_1^1 [HH96]	Δ_1^0	FSO
Pfad in Ordnungsbaum	Σ_1^1 [HH96]	Δ_1^0 [KRS05]	FSO
Mengenüberdeckung	Σ_1^1 [HH96]	Δ_1^0 [KL10]	FSO
Mengenpackung	Σ_1^1 [HH96]	Δ_1^0	FSO
Exakte Überdeckung	Σ_1^1 [HH96]	Σ_1^1	Pfad in Nachfolgerbaum
Erfüllbarkeit			
allgemein	Σ_1^1 [HH96]	Σ_1^1	Exakte Überdeckung
keine unendlichen Klauseln	vermutlich Π_2^0	Π_1^0	Post'sches Korrespondenzproblem
endlich viele unendliche Klauseln	vermutlich Σ_4^0	Σ_2^0	Variation PCP / TOTAL
Färbung			
unendlich viele Farben	Σ_1^1 [HH96]	Δ_1^0	Unabhängige Knotenmenge
endlich viele Farben	Π_1^0 [Bea76]	Π_1^0	Post'sches Korrespondenzproblem

Tabelle 7.1. Vollständigkeiten aller betrachteten Probleme im Vergleich zwischen rekursiven und automatischen Graphen sowie die hier verwendeten Reduktionen.

Literaturverzeichnis

[ADH98] Armen S Asratian, Tristan MJ Denley und Roland Häggkvist: *Bipartite graphs and their applications*. 131. Cambridge University Press, 1998.

[BE51] Nicolaas G de Bruijn und Paul Erdős: *A colour problem for infinite graphs and a problem in the theory of relations*. Indag. Math, Vol. 13 (5): Seiten 371–373, 1951.

[Bea76] Dwight R Bean: *Effective Coloration*. Journal of Symbolic Logic, Vol. 41 (2): Seiten 469–480, 1976.

[BG00] Achim Blumensath und Erich Grädel: *Automatic structures*. In: Logic in Computer Science, 2000. Proceedings. 15th Annual IEEE Symposium on, Seiten 51–62. IEEE, 2000.

[Blu99] Achim Blumensath: *Automatic Structures*. Diploma thesis, RWTH-Aachen, 1999.
URL `http://www.logic.rwth-aachen.de/pub/blume/AutStr.ps.gz`

[Göd31] Kurt Gödel: *Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I*. Monatshefte für Mathematik und Physik, Vol. 38 (1): Seiten 173–198, 1931.

[HH96] Tirza Hirst und David Harel: *Taking it to the limit: on infinite variants of NP-complete problems*. Journal of Computer and System Sciences, Vol. 53 (2): Seiten 180–193, 1996.

[Kar72] Richard M Karp: *Reducibility among combinatorial problems*. Springer, 1972.

[KL10] Dietrich Kuske und Markus Lohrey: *Some natural decision problems in automatic graphs*. Journal of Symbolic Logic, Vol. 75 (2): Seiten 678–710, 2010.

[KN95] Bakhadyr Khoussainov und Anil Nerode: *Automatic presentations of structures*. In: Logic and computational complexity, Seiten 367–392. Springer, 1995.

[Koz06] Dexter Kozen: *Theory of computation*. Springer, 2006.

[KRS05] BAKHADYR KHOUSSAINOV, SASHA RUBIN und FRANK STEPHAN: *Automatic linear orders and trees.* ACM Transactions on Computational Logic (TOCL), Vol. 6 (4): Seiten 675–700, 2005.

[Kus09] DIETRICH KUSKE: *Theories of automatic structures and their complexity.* In: Algebraic Informatics, Seiten 81–98. Springer, 2009.

[Mar04] DÁNIEL MARX: *Graph colouring problems and their applications in scheduling.* Electrical Engineering, Vol. 48 (1-2): Seiten 11–16, 2004.

[Pos46] EMIL L POST: *A variant of a recursively unsolvable problem.* Bulletin of the American Mathematical Society, Vol. 52 (4): Seiten 264–268, 1946.

[Rog67] HARTLEY ROGERS: *Theory of recursive functions and effective computability.* McGraw-Hill series in higher mathematics, 1967.

[Rub08] SASHA RUBIN: *Automata presenting structures: A survey of the finite string case.* Bulletin of Symbolic Logic, Vol. 14 (02): Seiten 169–209, 2008.

[Sch92] UWE SCHÖNING: *Theoretische Informatik - kurzgefasst.* BI Wissenschaftsverlag Mannheim, 1992.

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Schriften entnommen wurden, sind als solche kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen.

Ilmenau, den 10. September 2014

Chris Köcher